VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Testování simulačních prostředí síťových technologií

Benchmarking of Networking Simulation Environment

2017 Bc. Marek Večerka

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Marek Večerka**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Testování simulačních prostředí síťových technologií
Benchmarking the Networking Simulation Environment

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je srovnat existující simulační prostředí pro simulaci počítačových sítí, porovnat jejich vlastnosti, konfigurovatelnost a výkonnost dostupných řešení z hlediska funkcí 2.-7. vrstvy ISO-OSI modelu.

1. Seznamte se s technologiemi, běžně používanými k simulaci síťových technologií (GNS3, ns-3, apod.), a profesionálním řešením, založeným na platformě OpenStack (např. Cisco Virl).
2. Srovnejte možnosti konfigurace, poskytované platformou na 2.-7. vrstvě ISO-OSI modelu a způsoby, jak lze konfiguraci provádět.
3. Vytvořte scénář testů a příklady 2-3 typických simulovaných sítí.
4. Nasaďte simulované sítě ve vybraném simulátoru a profesionálním řešení a srovnejte paměťové a výpočetní nároky kladené na provoz obou řešení. Popište funkce, které je nutno konfigurovat nestandardním způsobem a případné části sítě, které nebylo možno simulovat.
5. Proveďte jednotlivé testy, které budou zaměřeny na propustnost sítě, možnosti zachycení provozu v síti, výkon jednotlivých síťových prvků a výkon celkového řešení. Výsledky testů vyhodnoťte.

Seznam doporučené odborné literatury:

[1] Tom Fifield, Diane Fleming, Anne Gentle, Lorin Hochstein, Jonathan Proulx, Everett Toews, Joe Topjian: OpenStack Operations Guide, O'Reilly Media, 2014, 330 stran, ISBN: 978-1-4919-4694-7. Dostupné z: <http://docs.openstack.org/ops/>.
[2] Jason C. Neumann: The Book of GNS3: Build Virtual Network Labs Using Cisco, Juniper, and More. No Starch Press, 2015, 272 stran, ISBN: 978-1-59327-695-9.
[3] Chris Welsh: GNS3 Network Simulation Guide. Packt Publishing, 2013, 154 stran, ISBN: 978-1-78216-081-6
[4] Cisco VIRL [online] [cit. 2016-02-15]. Dostupné z: <http://virl-dev-innovate.cisco.com/>

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Pavel Moravec, Ph.D.**

Datum zadání: 01.09.2016

Datum odevzdání: 28.04.2017

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 28. dubna 2017 .....

Rád bych touto cestou poděkoval Ing. Pavlu Moravcovi, Ph.D. za vedení mé diplomové práce, cenné rady a odborný dohled. Také děkuji všem, kteří mě podporovali během studia a vzniku této diplomové práce.

Abstrakt

Cílem práce je tvorba scénářů popisujících chování typických simulovaných sítí, jejich nasazení v těchto simulačních prostředích, a srovnání možností konfigurace simulovaných prvků na 2.-7. vrstvě ISO – OSI. Dále jsou nad vytvořenými scénáři provedeny testy zaměřené na paměťové a výpočetní nároky obou prostředí, propustnost sítě, možnosti zachycení provozu v síti a výkon jednotlivých síťových prvků resp. celkového řešení.

Hlavním přínosem práce je vyhodnocení jednotlivých testů a z něj vyplývající porovnání simulačního prostředí GNS3 s profesionálním řešením VIRL.

Klíčová slova: OpenStack, VIRL, GNS3, simulace, benchmarking, iPerf

Abstract

The aim of the thesis is to create scenarios describing the behavior of typical simulated networks, their deployment in these simulation environments, and comparing the configuration options of simulated elements to 2.-7. Layer ISO – OSI. In addition, above-mentioned scenarios are performed tests focused on memory and computational demands of both environments, network throughput, network capture capabilities and performance of individual network elements, respectively of the overall solution.

The main benefit of this thesis is the evaluation of the individual tests and the resulting comparison of the simulation environment GNS3 with the professional solution VIRL.

Key Words: OpenStack, VIRL, GNS3, simulation, benchmarking, iPerf

Obsah

Seznam použitých zkratek a symbolů

ABR	–	Area Border Router
API	–	Application Programming Interface
AS	–	Autonomní systém
AToM	–	Any Transport over Multiprotocol Label Switching
BGP	–	Border Gateway Protocol
BIOS	–	Basic Input-Output System
BSD	–	Berkeley Software Distribution
CCIE	–	Cisco Certified Internetwork Expert
CCNA	–	Cisco Certified Network Associate
CCNP	–	Cisco Certified Network Professional
CE	–	Customer Edge
CPU	–	Central Processing Unit
DHCP	–	Dynamic Host Configuration Protocol
EBGP	–	External Border Gateway Protocol
EIGRP	–	Enhanced Interior Gateway Routing Protocol
GUI	–	Graphical User Interface
HSRP	–	Hot Standby Router Protocol
HW	–	Hardware
IaaS	–	Infrastructure as a Service
IBGP	–	Internal Border Gateway Protocol
IGP	–	Interior Gateway Protocol
IOS	–	Internetwork Operating System
IOU	–	IOS on UNIX
IP	–	Internet Protocol
IPv4	–	Internet Protocol version 4
IPv6	–	Internet Protocol version 6
LDP	–	Label Distribution Protocol
LTS	–	Long Term Support
MP – BGP	–	Multiprotocol BGP
MPLS	–	Multiprotocol Label Switching
NASA	–	National Aeronautics and Space Administration
NAT	–	Network Address Translation
OSPF	–	Open Shortest Path First
PaaS	–	Platform as a Service
PC	–	Personal Computer
PE	–	Provider Edge

PID	–	Process ID
PVST	–	Per – VLAN Spanning Tree
QEMU	–	Quick Emulator
RAM	–	Random Access Memory
RD	–	Route Distinguisher
RIP	–	Routing Information Protocol
SaaS	–	Software as a Service
SDN	–	Software – defined networking
STP	–	Spanning Tree Protocol
TCP	–	Transmission Control Protocol
UDP	–	User Datagram Protocol
vCPU	–	virtual Central Processing Unit
VIRL	–	Virtual Internet Routing Lab
VLAN	–	Virtual Local Area Network
VPCS	–	Virtual Personal Computer Simulator
VPN	–	Virtual Private Network
VPNv4	–	Virtual Private Network version 4
VRF	–	Virtual Routing and Forwarding
WAN	–	Wide Area Network

Seznam obrázků

Seznam tabulek

1 Úvod

Simulační prostředí umožňují uživatelům na svých počítačích pohodlně navrhovat a konfigurovat síťové technologie bez potřeby vlastnit fyzický hardware. K tomuto účelu je možné použít open-source simulační prostředí GNS3, nebo si zakoupit licenci na profesionální řešení VIRL za $ 199.99 na rok. Cílem této práce je právě porovnat na jedné straně nekomerční simulační prostředí GNS3 a na straně druhé VIRL, komerční profesionální řešení.

Hlavní motivace proč srovnávat tyto dvě simulační prostředí je zjistit, jestli placené profesionální řešení VIRL od firmy Cisco převyšuje kvality a možnosti nabízené simulačním prostředí GNS3. V této práci chci navrhnout a nakonfigurovat tři příklady typických simulovaných sítí v obou simulačních prostředí a tyto prostředí porovnat z hlediska možnosti konfigurace jednotlivých navržených sítí.

Dále chci provést srovnání obou simulačních prostředí na základě paměťových a výpočetních nároků kladených na provoz, během spuštění simulací jednotlivých navržených sítí. Z hlediska zachycení provozu mě zajímá, jakými možnostmi pro zachycení provozu disponují jednotlivé simulační prostředí. A v neposlední řadě, chci na navržených sítích provést sadu testů, které budou zaměřeny na přenosovou rychlost uvnitř sítě, ztrátu UDP datagramů během přenosu, testování TCP spojení a dobu odpovědi na ICMP zprávy.

Simulační prostředí GNS3 je v této práci nasazené na operačním systému Linux a Windows. K profesionálnímu řešení jsem měl přístup v rámci univerzity.

V kapitole 2 je představen framework OpenStack, jakožto platforma na které je založené profesionální řešení VIRL. Součástí této kapitoly je stručná historie tohoto projektu a dále jsou zde popsány možnosti použití OpenStacku v oblasti virtualizace.

Simulační prostředí GNS3 je popsáno v kapitole 3. V této kapitole je čtenář seznámen s hlavními rysy a omezením tohoto prostředí. Dále je popsán způsob instalace GNS3 ze zdrojových souborů na operačním systému Linux Ubuntu 16.04 LTS Xenial Xerus.

V kapitole 4 je čtenář seznámen s profesionálním řešením VIRL a jeho hardwerovými požadavky na provoz. V této kapitole je také popsáno VM Maestro, jakožto aplikace na straně klienta, která umožňuje práci s prostředím VIRL. Součástí kapitoly jsou také dostupné zařízení po zakoupení licence na profesionální řešení VIRL.

Navržené scénáře typických simulovaných sítí jsou uvedeny v kapitole 5. Součást každého scénáře je stručný úvod a použité technologie v daném scénáři. Dále každý scénář obsahuje popsanou konfiguraci, schéma a adresní plán.

Kapitola 6 je věnovaná možnostem zachycení provozu v simulační platformě GNS3 a profesionálním řešení VIRL.

Testování jednotlivých scénářů pomocí nástrojů iPerf a PING jsou uvedeny v kapitole 7. Dále kapitola obsahuje testy na paměťové a výpočetní nároky nasazení scénářů na obou simulačních platformách. K tomuto účelu byl vytvořen testovací skript ve skriptovacím jazyku PowerShell.

Jednotlivé testy obsahují popis testu, naměřené hodnoty a jejich grafické znázornění, vyhodnocení.

V závěru mé práce, v kapitole 8 jsou vyhodnoceny získané poznatky během testování jednotlivých navržených scénářů.

2 OpenStack

V této kapitole, která si klade za cíl seznámit čtenáře s OpenStackem, jsem čerpal ze zdrojů [1] a [2].

OpenStack je možno definovat jako framework[1], pro řízení, definování a využívání cloudových[2] zdrojů. Oficiální stránka[3] OpenStacku popisuje framework jako „otevřený software, pro vytváření soukromých a veřejných cloudů.“a dále dodává, „OpenStack software přináší cloudový operační systém, který je masivně škálovatelný“.

2.1 Historie

Americký prezident Barack Obama během jeho prvního dne v úřadu, roku 2009 podepsal memorandum, které mělo prolomit bariéry a umožnit vetší spolupráci mezi vládou a lidmi jimž slouží. Memorandum je známé jako Open Government Directive.

Sto dvacet dní od vydání směrnice, NASA oznámila svůj Open Government framework, který představil nástroj nazvaný Nebula[4]. Nebula byla vytvořena pro rychlejší dodávání IaaS zdrojů mezi vědci z NASA a výzkumníky. Ve stejné době, poskytovatel cloud computingu firma Rackspace oznámila open – source objektové úložiště nazvané Swift.

V červenci 2010, Rackspace a NASA, spolu s dalšími 25 společnostmi, zahájili projekt OpenStack. Během pěti let, OpenStack byl vydán ve více než 10 verzích.

OpenStack udržuje šestiměsíční uvolňovací cyklus, který je koordinován s OpenStack summity. Projekt se rozrostl z 25 zúčastněných společností na více než 200, s tisíci uživateli z více než 130 zemí.

2.2 Virtualizační framework

Pokud má čtenář zkušenosti v oblasti virtualizaci serverů, může snadno dojít k závěru, že OpenStack je jen další způsob jak poskytovat virtuální stroje. Přestože OpenStack umožňuje poskytovat virutální stroje, v žádném případě se nejedná o jeho definitivní funkci.

Obrázek 2 ilustruje několik zdrojových komponent, které OpenStack koordinuje k vytváření veřejných a privátních cloudů. Jak je na obrázku znázorněno, OpenStack nenahrazuje zdroje poskytovatelů, ale jednoduše je spravuje prostřednictvím řídících bodů zabudovaných do frameworku.

Zkušený systémový administrátor může přijmout definici Openstacku jako cloudový operační systém se značnou nedůvěrou. Administrátor totiž neběží kolem stovky serverů s bootovacími disky a nenahrává OpenStack na holý kov, jako tradiční operační systém. Nicméně, prostřed-

[1] Skutečná, nebo konceptuální struktura, která slouží jako podpora, nebo návod pro stavbu něčeho co rozšiřuje strukturu do něčeho užitečného.

[2] Jakýkoliv zdroj, který je dostupný prostřednictvím počítačové sítě, resp. Internetu.

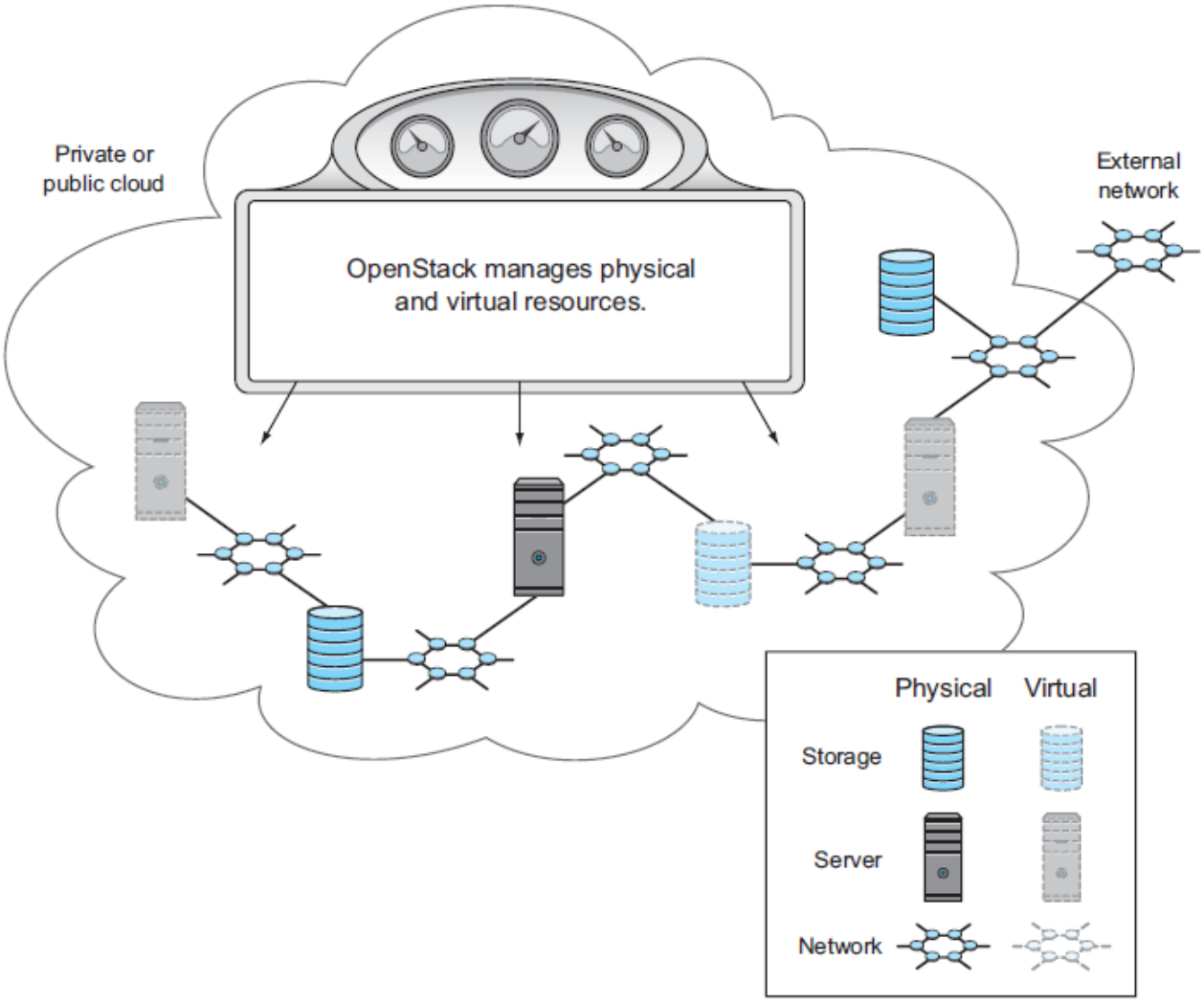
[3] https://www.openstack.org

[4] https://www.nebula.com/home/

nictvím řízení zdrojů, OpenStack sdílí operace charakterizující systémy, ale v souvislosti s cloud výpočty.

OpenStack umožňuje:

- Využívat zdroje fyzických a virtuálních serverů, sítí a úložných systémů,
- efektivní správa zdrojů cloudu, prostřednictvím nájemníků/tenants/, kvót a uživatelských rolí,
- poskytuje společné rozhraní k řízení zdrojů, bez ohledu na dodavatele subsystému.



Obrázek 1: OpenStack jako cloudový operační systém [1].

Na první pohled OpenStack nevypadá jako tradiční operační systém, ale potom je nutné si také uvědomit, že cloud nevypadá jako normální počítač.

OpenStack a cloudová terminologie

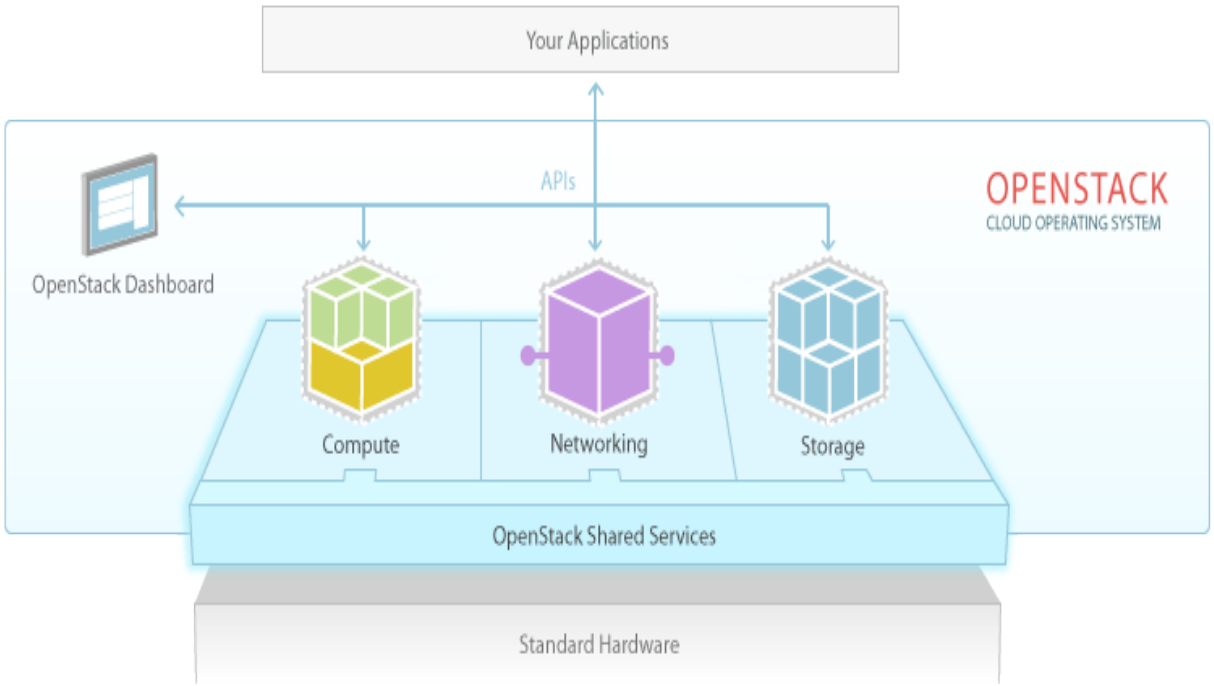
OpenStack je využíván pro konstrukci veřejných, nebo privátních cloudů. Z pohledu veřejných

a privátních cloudů můžeme také hovořit o cloudu jako určité poskytované službě. OpenStack je tedy využíván jako základ pro poskytnutí cloudu jako služby.

Předpokládejme, že uživatel má zájem o poskytnutí VM, nebo prostředků pro ukládání dat u Amazonu. V tomto případě hovoříme o infrastruktuře jako službě (IaaS). Uživatel má přímý přístup k poskytnutému virtuálnímu stroji, který může přímo spravovat. Fyzické komponenty, které tvoří cloud, jsou před uživatelem skryty a virtuální komponenty jsou přímo přístupné. OpenStack je odpovědný za řízení zdrojů, které jsou poskytovány koncovým uživatelům.

Nyní předpokládejme, že uživatel vlastnící cloud nemá přímý přístup k IaaS, ale puze k aplikaci prostřednictvím funkcí, které poskytuje OpenStack. V tomto kontextu může být OpenStack považován za back end nabízené platformy jako služby (PaaS). Veškerá fyzická a virtuální infrastruktura je skryta před uživatelem. Uvažujme případ, kdy tým vývojářů potřebuje izolované aplikační prostředí (aplikační vrstva nasazená na IaaS) pro testování software. Prostřednictvím cloud orchestaci může být OpenStack na back endu v nasazené testovací platformě.

Nyní předpokládejme, že naše firma poskytuje služby svým zákazníkům pomocí IaaS, nebo PaaS prostřednictvím OpenStacku. V tomto kontextu OpenStack slouží jako back end komponenta software jako služby (SaaS). Jak si můžeme všimnout, OpenStack může být použit jako základní komponenta v mnoha různých vrstvách cloud computingu.



Obrázek 2: Softwérový diagram OpenStacku [2].

2.3 Základní komponenty

V následující tabulce jsou uvedeny OpenStack komponenty, nebo důležité projekty. OpenStack obsahuje daleko více projektů v různých fázích vývoje, ale v tabulce jsou záměrně uvedeny pouze základní komponenty OpenStacku. Aktuální služby, které poskytuje OpenStack jsou uvedeny na oficiálnách stránkách v sekci RoadMap[5].

Tabulka 1: Základní komponenty OpenStacku.

Projekt	**Kódové označení**	**Popis**
Compute	Nova	Správa VM prostředků, včetně CPU, paměti, disku a síťového rozhraní.
Networking	Neutron	Proskytuje zdroje využívané VM síťového rozhraní, všetně IP adresování, směrování a softwarově definovaného síťování (SDN).
Object Storage	Swift	Poskytuje úložiště na úrovni objektů, přístupné prostřednictvím RESTful API.
Block Storage	Cinder	Poskytuje tradiční diskové úložiště pro VM.
Identity	Keystone	Spravuje řízení přístupu pro komponenty OpenStacku. Poskytuje autorizační služby.
Image Service	Glance	Spravuje obrazy disků VM. Dodává obrazy disků pro VM a slouží pro okamžité zálohy.
Dashboard	Horizon	Poskytuje webové GUI pro práci s OpenStackem.
Telemetry	Ceilometer	Poskytuje sbírku pro měření a monitorování OpenStack komponent.
Orchestration	Heat	Poskytuje cloudové aplikace ve formě šablon, pro prostředí OpenStacku.

[5]https://www.openstack.org/software/roadmap

2.4 Shrnutí

- IaaS cloudy tvoří sbírku komoditních zdrojů, které jsou koordinovány prostřednictvím frameworku.
- OpenStack je framework, který umožňuje koncovým uživatelům koordinovat IaaS a aplikační orchestraci (PaaS/SaaS).
- OpenStack řídí existující komerční a komunitní technologie jako hypervisory, úložiště, síťový hardware a software.
- OpenStack se skládá z kolekce projektů, kde každý projekt má svůj specifický účel.
- Každý projekt OpenStacku má přiděleno kódové označení.

3 GNS3

V této kapitole, kde je čtenáři představena simulační platforma GNS3 jsem čerpal ze zdrojů [7], [8] a [9].

GNS3 patří mezi nejoblíbenější síťové simulátory a je běžně používán studenty, kteří potřebují získat praktické zkušenosti s Cisco IOS směrováním a přepínám, během přípravy na Cisco Certified Network Associate (CCNA) a Cisco Certified Network Professional 2 část 1 (CCNP). Dále tento software používají organizace jako je Walmart, Exxon, Twitter, AT&T, NASA a ministerstvo obrany Spojených států amerických.

GNS3 je open – source grafický síťový simulátor, který uživateli umožňuje navrhnout a testovat virtuální sítě přímo na jeho PC. Tento simulační nástroj dokáže emulovat několik systémů, včetně směrovačů od firem jako je Cisco, Juniper, MikroTik, Arista a Vyatta. V GNS3 je také možné emulovat Linux a Windows virtuální stroje.

Výhoda GNS3 spočívá v možnostech použití virtualizace. Žijeme ve světě, kde je čím dál větší snaha zařízení virtualizovat a minimalizovat potřebu vlastnit fyzický hardware. Tento předpoklad splňuje GNS3 dokonale. Dokud nebyla dostupná virtualizace, studenti, síťoví inženýři, administátoři museli vybudovat laboratoře s potřebným hardwarem, nebo si tyto laboratoře pronajímat. Obě možnosti mohou být drahé, nepohodlné a zároveň uživatele do jisté míry omezují na základě dostupného hardwaru. GNS3 umožňuje uživataeli přizpůsobit jeho síťové laboratoře, aby přesně vyhovovaly jeho potřebám.

Dále GNS3 poskytuje návrhům uživatele maximální flexibilitu prostřednictvím kombinace emulovaných hardwarových zařízení, na kterých běží reálné síťové operační systémy, jako je Cisco IOS, nebo simulované operační systémy jako je NX – OSv a také možnost sdílet prostředky mezi více počítači.

3.1 Hlavní rysy GNS3

Emulace hardware

Grafické uživatelské rozhraní GNS3 umožňuje pohodlně vytvářet virtualizované síťové laboratoře s různými směrovači, přepínači a PC, ale opravdovou sílu GNS3 uživatel pocítí až v momentě, kdy použije ve své laboratoři Cisco IOS. Na rozdíl od podobných aplikací, GNS3 uživatele neomezuje napodobováním příkazů, nebo funkcí Cisco IOS. Namísto toho využívá hypervisor, který emuluje hardware, na kterém běží Cisco IOS. Emulace hardware je v případě GNS3 velice zásadní, neboť nám umožňuje spouštět aktuální image konkrétního IOS přímo na našem PC.

Veškeré konfigurační příkazy a jejich výstupy vychází z reálných IOS a tedy můžeme říct, že veškeré protokoly, nebo funkce podporované zvolenou verzí IOS jsou k dispozici, pro použití v našich navržených modelech sítí.

Touto vlastností se GNS3 liší od ostatních programů, jako je RouterSim, Boson NetSim, nebo VIRL, které pouze simulují zařízení a poskytují tedy omezené prostředí IOS, konfigurační pří-

kazy a tedy i možné scénaře nasazení.

Simulované operační systémy
Kromě emulace hardware GNS3 integruje simulované operační systémy, které mohou být propojeny s ostatním GNS3 zařízením. Příkladem může být Cisco IOU, cože je vpodstatě IOS pro Unix. IOU se skládá z několika binárních souborů pro Linux, které emulují vlastnosti klasického Cisco IOS.

Dále je možné do GNS3 integrovat QEMU a obrazy virtuálních systémů z VirtualBoxu, jako např. Linux, BSD, nebo Windows. Například pro nasazení Apache web serveru na Linuxu stačí pouze přidat virtuální stroj, na kterém běží Linux a Apache. Následně tento VM vložit do GNS3, připojit k okolním zařízením a můžeme testovat prohlížení webu z jiného VM. Vše v rámci uživatelského prostředí GNS3.

Využití open – source technologií
Jak již bylo řečeno, GNS3 využívá open source technologie jako jsou Dynamips, QEMU a Virtualbox. Díky těmto technologiím můžeme spouštět operační systémy od společností jako jsou Juniper, Arista a mnoho dalších síťových operačních systémů stejně snadno jako je tomu v případě Cisco IOS.

Dynamips Hypervisor
Pro emulování Cisco hardwaru používá GNS3 aplikaci Dynamips. Tato aplikace byla vytvořena v roce 2005 a její autor je Christophe Fillot z Franice a v chodu je udržována Fláviem J. Saraivem a dalšími. Dynamips hypervisor může emulovat Cisco router hardware v sériích 1700, 2600, 3600, 3700 a 7200.

Díky aplikaci Dynamips můžeme snadno a rychle konfigurovat tyto modely směrovačů s různými emulovanými Cisco síťovými sloty a WAN rozhraním. Virtuální vstupně / výstupní karty umožňují přidat k našim zařízením více ethernetových rozhraní, switchovací moduly a sériové porty. Dokonce je možné nastavovat paměť jednotlivým zařízením, na základě požadavků použité verze Cisco IOS.

QEMU a VirtualBox
Uživatel může ve svých proktech používat virtuální stroje z virtualizačních platforem jako je VirtualBox, nebo QEMU. Virtuální stroje mohou být následně spojeny s jinými GNS3 zařízeními pro sítě typu end – to – end.

3.2 Limity GNS3

Ač by se mohlo zdát, že GNS3 je dokonalá simulační platforma, není tomu tak! I GNS3 má své limity.

Například, Dynamips byl omezen takovým způsobem, že jej není možné použít v produkčním prostředí, ale pouze pro vzdělávací účely. Dynamips není schopen emulovat integrovaný obvod ASIC, který je součástí pokročilých Catalyst přepínačů od firmy Cisco a z toho plyne, že přepínáni v GNS3 je značně omezeno. Nicméně, GNS3 umožňuje integrovat do virtuální laboratoře skutečný přepínač řady Catalyst a je možné využívat veškeré přepínací funkce.

Pokud ale uživatel nepotřebuje nutně pokročilé funkce přepínače, vystačí si s přidáním virtuálního přepínacího modulu jako je Cisco NM – 16ESW a z vašeho virtuálního směrovače vytvoří jedoduchý přepínač na 3. vrstvě. Takto nastavené zařízení splňuje běžné přepínací potřeby. Podporuje následující technologie: VLAN, 802.1Q trunking, spanning – tree, EtherChannel a multiprotokolové směrování pomocí EIGRP, OSPF, BGP a ostatní protokoly. Navíc je tu možnost použít Cisco IOU, které mohou být použity k emulaci přepípačů Cisco a poskytují více příkazů, než je tomu v případě spínacího modulu.

Další omezení spočívá ve výkonu sítě. Vzhledem k tomu, že Dynamips je emulátor, který neposkytuje žádnou hardwarovou akceleraci, propustnost sítě je omezena na 1,5Mb až 800Mb za sekundu, vše záleží na použitém IOS a konfiguraci. Na první pohled se tato vlastnost může zdát jako omezení, ale není tomu tak úplně, je to vlastně dobrá věc, protože zabraňuje uživatelům umístit virtualizovaný Cisco hardware do produkčního prostředí. Představme si, že by emulované zařízení běželo na plné propustnosti, potom by bylo snadé do každé síťe nainstalovat virtuální směrovač na levný PC a použít je v produkčních sítích, což by způsobovalo krádež duševního vlastnictví společnosti Cisco.

Firma Cisco nabízí alternativu k GNS3, ve formě produktu s názvem Virtual Internet Routing Lab (bude představen čtenáři v následující kapitole). VIRL je velice podobný GNS3, ale vyžaduje roční poplatek oproti GNS3, která je zdarma. Zde je ovšem nutno podotknout, že GNS3 jako nástroj, který dokáže emuluvat různé systémy je sice zdarma (není vyloučené zpoplatnění v budoucnu), ale tyto systémy nejsou součástí GNS3 a je na každém uživateli, jak si je obstará. IOS je možné skopírovat z fyzického routeru, nebo prostřednictvím připojení Cisco online účtu, pokud máte smlouvu se společností Cisco.

Na závěr je nutné říct, že výše zmíněné omezení mají malý, nebo žádný účinek při použití GNS3 pro vzdělávací, nebo testovací účely.

3.3 Instalace

Instalace GNS3 pro OS Windows probíhá pomocí běžného instalačního souboru, proto zde tuto instalaci nebudu popisovat. Popsaná instalace byla provedena na operačním systému Linux Ubuntu 16.04 LTS Xenial Xerus.

Nejdříve si stáhneme z adresy https://github.com/GNS3/gns3-gui/releases/ požadovanou verzi GNS3. V této diplomové práci byla použita GNS3 ve verzi 1.5.2. Stažený archiv GNS3-1.5.2.source.zip obsahuje následující soubory: vpcs-0.6.1.zip, ubridge-0.9.4.zip, iouyap-0.97.zip, gns3-server-1.5.2.zip, gns3-gui-1.5.2.zip, dynamips-0.2.16.zip. Archiv extrahujeme a můžeme se pustit do instalace potřebných závislostí pro instalaci GNS3 a následně samotné instalace GNS3 ze zdrojových souborů.

Aktualizace zdrojových souborů a instalace závislostí GNS3.

```
sudo apt-get update
sudo apt-get install python3-dev
sudo apt-get install python3-setuptools
sudo apt-get install python3-pyqt5
sudo apt-get install python3-pyqt5.qtsvg
sudo apt-get install python3-pyqt5.qtwebkit
sudo apt-get install python3-ws4py
sudo apt-get install python3-netifaces
sudo apt-get install python3-pip
```

Instalace požadovaných závislostí pro kompilaci aplikace Dynamips.

```
sudo apt-get install cmake
sudo apt-get install uuid-dev
sudo apt-get install libelf-dev
sudo apt-get install libpcap-dev
```

Kompilace a instalace aplikace Dynamips.

```
unzip dynamips-0.2.16.zip
cd dynamips-0.2.16/
mkdir build
cd build/
cmake ..
make
sudo make install
sudo setcap cap_net_admin,cap_net_raw=ep /usr/local/bin/dynamips
cd ../..
```

Instalace GNS3 Serveru a GUI GNS3.

```
unzip gns3-server-1.5.2.zip
cd gns3-server-1.5.2/
sudo python3 setup.py install
cd ..
unzip gns3-gui-1.5.2.zip
cd gns3-gui-1.5.2/
sudo python3 setup.py install
```

Instalace závislostí IOU.

```
sudo apt-get install libssl1.0.0:i386
sudo ln -s /lib/i386-linux-gnu/libcrypto.so.1.0.0 /lib/libcrypto.so.4
sudo apt-get install bison
sudo apt-get install flex
sudo apt-get install git
git clone http://github.com/ndevilla/iniparser.git
cd iniparser/
make
sudo cp libiniparser.* /usr/lib/
sudo cp src/iniparser.h /usr/local/include
sudo cp src/dictionary.h /usr/local/include
```

Kompilace a instalace IOUYAP - aplikace umožňuje přemostění mezi UDP, TAP (virtuální Linuxové porty) a Ethernetem.

```
unzip iouyap-0.97.zip
cd iouyap-0.97/
sudo make install
sudo cp iouyap /usr/local/bin
```

Instalace ostatního software pro GNS3. Program cpulimit slouží pro omezení vytížení CPU. VirtualBox je známá virtualizační platforma, která umožňuje v GNS3 virtualizovat různé OS v roli hostů. QEMU je open-source emulátor hardwarové a softwarové virtualizace. Wireshark umožňuje zachytávat zvolený provoz v naší virtuální laboratoři GNS3.

```
sudo apt-get install cpulimit
sudo apt-get install virtualbox
sudo apt-get install qemu
sudo apt-get install wireshark
```

Kompilace a instalace VPCS.

```
unzip vpcs-0.6.1.zip
cd vpcs-0.6.1/src
./mk.sh
sudo cp vpcs /usr/local/bin/
cd ../..
```

Instalace aplikace uBridge. Jedná se o jednoduchou aplikaci, která umožňuje vytvářet mosty mezi různými technologiemi. V současné verzi uBridge je možné přemostění mezi UDP tunely, Ethernetem a TAP rozhraním. Dále je možné provádět zachytávání paketů. Aplikace uBridge je podobná aplikaci IOUYAP a v GNS3 je z důvodu podpory VMware.

```
unzip ubridge-0.9.4.zip
cd ubridge-0.9.4/
make
sudo make install
```

4 VIRL

Virtual Internet Routing Lab je rozšiřitelná síťová virtualizační platforma, která umožňuje vývoj vysoce důvěryhodných modelů skutečných, nebo plánovaných sítí. VIRL zahrňuje aktuální virtualizazované Cisco síťové operační systémy a umožňuje integraci se skutečnými fyzickými / externími sítěmi, síťovými prvky a servery. Práce ve VIRL je možná prostřednictvím VM Maestra, nebo přímo na serveru, na kterém je nainstalovaný VIRL prostřednictvím webového rozhraní [3].

4.1 HW požadavky

- Hostitelský systém musí mít přístup do Internetu (TCP porty 4505 a 4506 odchozí povoleny na firewallu/ proxy).
- Minimálně 8 GB RAM a čtyři jádra CPU musí být alokovány k virtuálnímu stroji VIRL pro lepší výkon a funkčnost. Pro velké simulace se doporučuje více zdrojů a to např. pro 20 uzlů 12GB, 30 uzlů 15GB, 40 uzlů 18GB. Cisco IOSXRv (reprezentuje IOSXR[6] včetně software a operačního systému) vyžaduje pro spuštění 3GB paměti.
- Intel VT – x/ EPT nebo AMD – V/ RVI včetně virtualizačního rozšíření povoleného v BIOSU.
- 70GB volného místa na disku pro instalaci.

Zde prezentované HW požadavky jdou udávány přímo firmou Cisco [4].

4.2 VM Maestro

Pro popis této kapitoly, věnované VM Maestru byl použit zdroj [5]. VM Maestro je aplikace spustitelná na straně klienta, která poskytuje vývojové prostředí pro vytvoření a vizualizaci topologie a její konfigurace. Dále VM Maestro umožňuje spravovat simulace, které jsou vykonávány na hostitelské VIRL instaci, nebo virtuálním stroji.

Aplikace VM Maestro je dodávána společně s VIRL a je k dispozici pro instalaci na operačních systémech Windows, OS X a Linux. VM Maestro je dostupné na IP adrese hostovaného VIRL, nebo virtuálního stroje, kde je VIRL nainstalovaný.

Instalace a konfigurace

Instalace, respektive spuštění VM Maestro klienta je velice snadná.

1. Uživatel vloží do prohlížeče adresu serveru, na kterém je nainstalovaný VIRL a v nabídce zvolíme položku VM Maestro Clients.

[6] Síťový operační systém pro sítě poskytovatelů služeb od firmy Cisco.

2. Následně stáhneme klienta VM Maestro, který odpovídá naší platformě. Stáhneme tedy klienta pro operační systém Windows. Nyní stačí archiv rozbalit a spustit VM Maestro. Archiv pro operační systém Windows obsahuje standartní instalační soubor.

3. Po spuštění klienta si přečteme a potvrdíme licenční podmínky.

4. Následně jsme vyzvání pro zadání IP adresy serveru, na kterém je nainstalovaný VIRL.

5. Zvolíme OK pro dokončení konfigurace VM Maestro klienta.

6. Poslední krok pro korektní používání VM Maestro klienta je stáhnutí všech typů uzlů, které jsou k dispozici na serveru, na kterém je nainstalovaný VIRL. V nabídce zvolíme FILE → Preferences → Node Subtypes. Zvolíme Restore Defaults a potvrdíme OK, poté aplikujeme provedené změny pomocí Apply. Dále zvolíme Fetch from Server a potvrdíme OK. Nakonec potvrdíme ještě jednou OK a máme hotovo. Nutno podotknout, že tuto procedůru je nutné provést pokaždé, dojde – li k přidání nových typů do VIRL hostitele.

4.3 Dostupné zařízení

Práce ve VM Maestro probíhá formou drag and drop, z tohoto důvodu budeme potřebovat nějaké zařízení. V následujících tabulkách je přehled základních zařízení, které jsou dostupné pro práci s VIRL. Zpracováno podle [6].

Směrovače

- IOSv - Směrovač s odlehčeným IOS, který podporuje veškeré potřeby pro CCNA / CCNP certifikaci.
- CSR1000v - Směrovač vhodný pro studium CCIE.
- IOS XRv - Vhodné pro certifikaci na Service Provider.

Přepínače

- IOSvL2 - Přepínač, který podporuje velké množství funkcí, které jsou vhodné pro certifikaci CCNA / CCNP / CCIE.
- Unmanaged Switch - Virutální přepínač velice podobný přepínači z GNS3. Slouží pouze pro propojení mezi zařázeními.
- NX – OSv - Přepínač Nexus, který podporuje funkce L2, L3 a některé specifické funkce tohoto přepínače.

Firewall

- ASAv - ASA firewall, který podporuje většinu funkcí pro CCNA Security certifikaci.

Servery

- Server - Ubuntu 14.
- LXC - Odlehčený linuxový kontejner.
- LXC – iperf - Odlehčený linuxový kontejner, který obsahuje iperf.
- LXC – ostinato - Odlehčený linuxový kontejner, který obsahuje generátor provozu ostinato.
- LXC – routem - Linuxový kontejner se směrovacím injektorem.

5 Scénáře

V této kapitole jsem navrhl 3 typické scénáře, které jsou nasadil v prostředí GNS3 a VIRL. U každého navrženého scénáře je uveden popis, konfigurace, schéma a adresní plán. Pro adresní plány jednotlivých scénářů byly použity prefixy dle RFC 5735[7] (speciální použití IPv4 adres) a jsou dostupné čtenáři v jednotlivých scénářích. Při realizaci jednotlivých scénářů byly použity image od firmy Cisco, takže se konfigurace v prostředí GNS3 a VIRL neliší.

5.1 Scénář 1

Cílem navrženého scénaře bylo nakonfigurovat autonomní systém (AS100) poskytovatele, ve kterém je zajištěna konektivita pomocí protokolu OSPF a MPLS. K tomuto autonomnímu systému, který zde figuruje jako tranzitní, jsou připojeny dvě pobočky společnosti, ve schématu reprezentovány jako AS110 a AS120. Konektivitu mezi pobočkami přes autonomní systém poskytovatele zajišťuje protokol BGP. Schéma navržené topologie je znázorněno na obrázku 3 a adresní plán pro Scénář 1 je čtenáři k dispozici na obrázku 4.

V pobočce, která má označení AS110 zajišťuje vnitřní konektivitu protokol OSPF. Dále je zde nakonfigurován DHCP server pro přidělování adres jednotlivým stanicím. Záložní konektivitu v případě výpadku směrovačů, které jsou nakonfigurovány jako výchozí brány jednotlivým stanicím, zde zajišťuje protokol HSRP.

Pobočka označená jako AS120 se od AS110 příliš neliší. Jediný rozdíl je v použitém IGP protokolu. Vnitřní konektivitu zde zajišťuje protokol RIP ve verzi 2. Dále je zde nakonfugorvána služba DHCP a o zajištění záložní konektivity v případě výpadku výchozích bran se postará protokol HSRP.

5.1.1 Realizace scénáře 1

Směrování a aktivace MPLS v páteřní síti poskytovatele

V páteřní síti poskytovatele zajišťuje konektivitu mezi jednotlivými směrovači protokol OSPF. Nakonfigurujeme IP adresy jednotlivým směrovačům podle adresního plánu a spustíme směrovací protokol OSPF. Protokol OSPF spustíme na všech zařízení poskytovatele v AS100 a do oblasti 0 vypropagujeme Loopback rozhraní a rozhraní přímo připojené do páteřní sítě poskytovatele.

```
router ospf

interface loopback 0
ip address 203.0.113.40 255.255.255.255
ip ospf 1 area 0
exit
```

[7] https://tools.ietf.org/html/rfc5735

```
interface serial 0/1
ip address 203.0.113.9 255.255.255.252
ip ospf 1 area 0
exit
```

Výpis 1: Ukázka konfigurace IP adresy na rozhraní a zařazení rozhraní do směrovacího protokolu OSPF.

Spuštěním technologie MPLS na rozhraních směrovačů přímo připojených do páteřní sítě a volbou LDP protokolu zahájíme distribuci MPLS značek mezi směrovači uvnitř sítě poskytovatele. Loopback rozhraní zde poslouží jako router – id pro technologii MPLS.

```
interface serial 0/1
mpls ip
no shutdown

mpls label protocol ldp
mpls ldp router-id Loopback0 force
```

Výpis 2: Spuštění technologie MPLS a distribuce značek.

Směrování uvnitř poboček zákazníka

V pobočce označené jako AS110 zajišťuje konektivitu protokol OSPF. Nastavíme IP adresy rozhraním směrovačů, které jsou přímo připojené do sítě zákazníka a spustíme směrovací protokol OSPF. Nakonfigurované rozhraní vypropagujeme do oblasti 0 směrovacího protokolu OSPF.

```
router ospf 1
exit

interface s0/0
ip add 198.51.100.65 255.255.255.252
ip ospf 1 area 0
no shutdown
exit
```

Výpis 3: Spuštění směrovacího protokolu OSPF a propagace IP adresy rozhraní do OSPF.

Druhá pobočka, tedy AS120, využívá pro zajištění směrování uvnitř svého AS směrovací protokol RIP ve verzi 2. Nejdříve tedy nakonfigurujeme IP adresy rozhraním podle adresního plánu a následně spustíme směrovací proces a tyto rozhraní vypropagujeme do směrovacího protokolu RIPv2.

```
interface s0/0
ip add 192.0.2.65 255.255.255.252
no shutdown
exit

router rip
version 2
no auto-summary
network 192.0.2.64
network 192.0.2.76
network 192.0.2.68
exit
```

Výpis 4: Spuštění směrování pomocí protokolu RIP verze 2.

Zajištění konektivity mezi autonomními systémy

Konektivitu mezi poskytovatelem a autonomními systémy zákazníka se stará technologie BGP. Uvnitř páteřní sítě poskytovatele AS100 provozujeme IBGP mezi hraničními směrovači poskytovatele PE1 a PE2. Mezi směrovači PE a CE je nakonfigurovaná technologie EBGP. Nejdříve nakonfigurujeme IP adresy na rozhraní, které spojují PE a CE směrovače, abychom mohli navázat sousedství s AS zákazníka s poskytovatele. Po nakonfigurování rozhraní, můžeme aktivovat BGP a provést konfiguraci sousedský spojení. Podle čísla AS uvedeného za remote – as technologie BGP pozná, zda je dané spojení v rámci IBGP, nebo EBGP. Pro IBGP uvedeme jako zdrojové rozhraní Loopback.

```
PE1
router bgp 100
 redistribute connected
 neighbor 203.0.113.26 remote-as 110
 neighbor 203.0.113.29 remote-as 110
 neighbor 203.0.113.41 remote-as 100
 neighbor 203.0.113.41 update-source Loopback0
```

Výpis 5: Konfigurace BGP na směrovači poskytovatele PE1 v AS100.

Obdobně provedeme konfiguraci i na druhém hraničním směrovači zákazníka PE2 akorát navážeme spojení s AS120.

Na hraničním směrovači zákazníka je konfigurace podobná jako na PE směrovači. Navážeme sousedství s PE směrovačem pro provoz eBGP a také se sousedským směrovačem CE2, pro provoz iBGP.

```
CE1
router bgp 110
neighbor 198.51.100.66 remote-as 110
neighbor 203.0.113.25 remote-as 100
```

Výpis 6: Konfigurace BGP na směrovači zákazníka CE1 v AS110.

Na směrovačích zákazníka je nutné provést redistribuci ze směrovacího protokolu OSPF do BGP, aby naučené směrovací informace v AS110 byly dostupné pro stanice v AS120.

```
CE1
router bgp 110
redistribute ospf 1
```

Výpis 7: Redistribuce z OSPF do BGP v rámci AS110.

Obdobná konfigurace je v AS120 akorát s rozdílem, že redistribujeme směrovací informaci z protokolu RIPv2.

Redistribuce směrovací informace z BGP do IGP

Na hraničních směrovačích zákazníka (CE5 a CE6) vytvoříme dva access – listy, ve kterých definujeme subnety zákazníka z AS120, které mají být vypropagovány z BGP do OSPF v AS110 a následně vytvoříme route – map, kterou si pojmenujeme jako MYMAP a přiřadíme jí vytvořené access – listy se subnety zákaznika z AS120. Následně provedeme samotnou redistribuci na základě definované route – mapy.

```
access-list 1 permit 192.0.2.0 0.0.0.31
access-list 1 permit 192.0.2.32 0.0.0.31

route-map MYMAP permit 10
match ip address 1

router ospf 1
redistribute bgp 110 subnets route-map MYMAP
```

Výpis 8: Redistribuce z OSPF do protokolu BGP.

HSRP

Nejdříve nakonfigurujeme směrovač, u kterého požadujeme aktivní roli, tedy stanice budou posílat veškerý provoz primárně přes něj. Kromě adresy rozhraní mu přidělíme virtuální adresu brány. Prioritu neměníme, standartně je na hodnotě 100. Dále zajistíme, aby po opětovném naběhnutí z výpadku přebral směrovač opět svou aktivní roli.

```
interface FastEthernet1/0
ip address 198.51.100.1 255.255.255.224
standby ip 198.51.100.3
standby preempt
```

Výpis 9: Konfigurace aktivního směrovače pro technologii HSRP.

U směrovače, který slouží jako záložní, je konfigurace velmi podobná, akorát s rozdílem, že mu snížíme prioritu na 99, čímž se z něj stane záložní směrovač v případě výpadku primárního směrovače.

```
interface FastEthernet1/0
ip address 198.51.100.2 255.255.255.224
standby ip 198.51.100.3
standby priority 99
```

Výpis 10: Konfigurace záložního směrovače pro technologii HSRP.

Uvedený příklad konfigurace se vztahuje na pobočku zákazníka v AS110 pro VLAN10. V případě AS120 je konfigurace stejná, akorát s použitím jiných IP adres.

DHCP

Standartně je služba DHCP serveru na přepínačích a směrovačích zapnuta, ale pokud tomu tak není, provedeme její zapnutí.

```
service dhcp
```

Výpis 11: Zapnutí služby DHCP.

Po zapnutí DHCP serveru se můžeme pustit do samotné konfigurace. Aby bylo možné přidělovat adresy, musíme nejdříve vytvořit DHCP Pool pro VLAN 10 a 20, v našem případě použijeme označení net10 a net20. Dále specifikujeme adresu výchozí brány.

```
ip dhcp pool net10
network 198.51.100.0 255.255.255.224
default-router 198.51.100.3

ip dhcp pool net20
network 198.51.100.32 255.255.255.224
default-router 198.51.100.35
```

Výpis 12: Konfigurace DHCP.

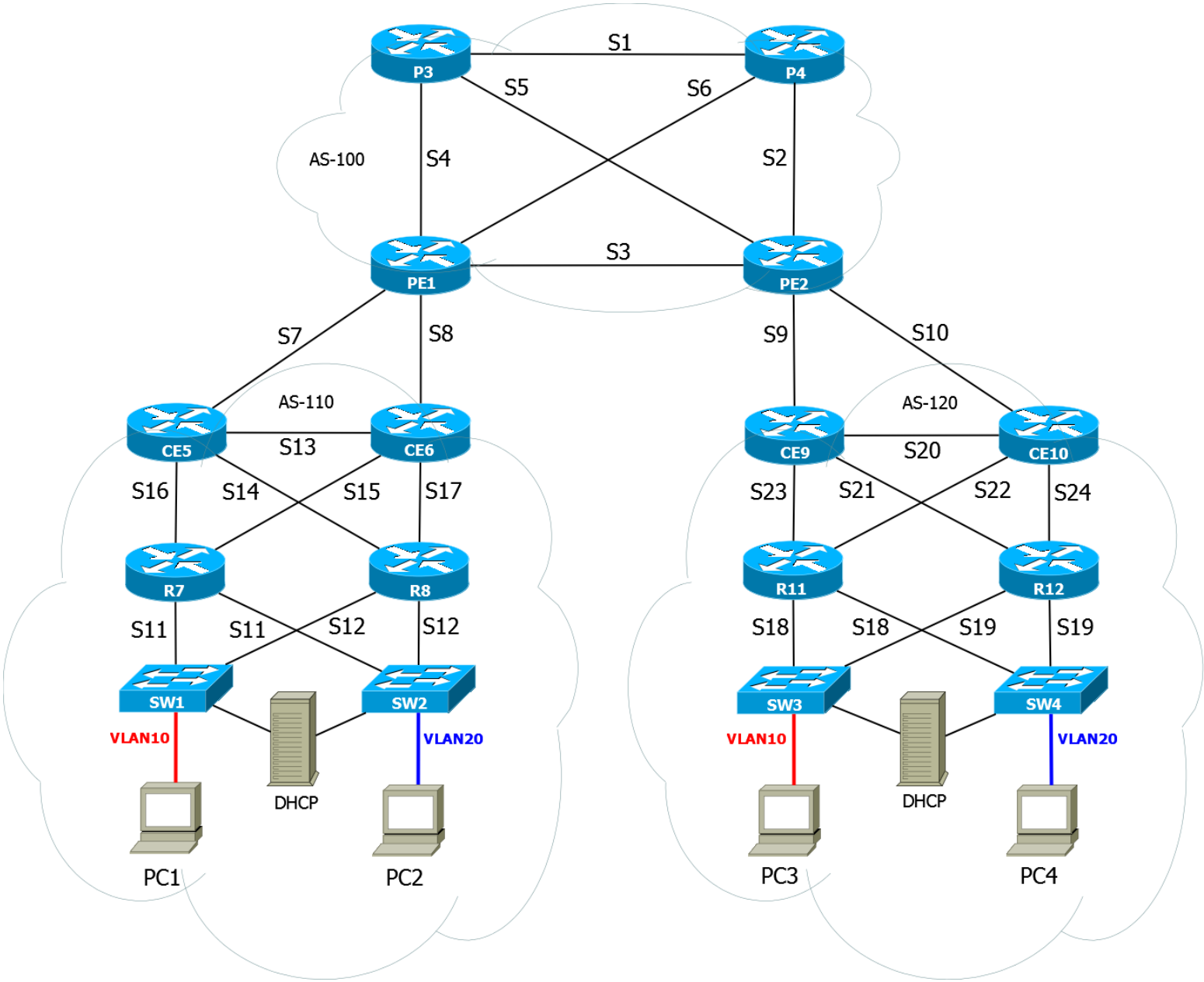
Vzhledem k tomu, že první 3 dostupné adresy ze zadaných rozsahů jsou využívány technologií HSRP a 4. adresa z poolu adres je použita pro rozhraní DHCP, chceme stanicím distribuovat adresy až od adresy 198.51.100.5, respektive 198.51.100.37. Musíme tedy provést vyloučení těchto adres pro přidělování.

```
ip dhcp excluded-address 198.51.100.0 198.51.100.4
ip dhcp excluded-address 198.51.100.32 198.51.100.36
```

Výpis 13: Vyloučení adres pro přidělování zákazníkům.

Konfigurace DHCP serveru je totožná u obou poboček, liší se pouze použitými adresami.

Kompletní výpisy konfigurací jednotlivých zařízení jsou k dispozici v příloze v adresáři Konfigurace/scenar1.



Obrázek 3: Schéma scénáře 1.

Síť poskytovatele : **203.0.113.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S1	4	203.0.113.0	203.0.113.1	203.0.113.2	203.0.113.3	/30
S2	4	203.0.113.4	203.0.113.5	203.0.113.6	203.0.113.7	/30
S3	4	203.0.113.8	203.0.113.9	203.0.113.10	203.0.113.11	/30
S4	4	203.0.113.12	203.0.113.13	203.0.113.14	203.0.113.15	/30
S5	4	203.0.113.16	203.0.113.17	203.0.113.18	203.0.113.19	/30
S6	4	203.0.113.20	203.0.113.21	203.0.113.22	203.0.113.23	/30
S7	4	203.0.113.24	203.0.113.25	203.0.113.26	203.0.113.27	/30
S8	4	203.0.113.28	203.0.113.29	203.0.113.30	203.0.113.31	/30
S9	4	203.0.113.32	203.0.113.33	203.0.113.34	203.0.113.35	/30
S10	4	203.0.113.36	203.0.113.37	203.0.113.38	203.0.113.39	/30

Loopback rozhraní	Adresa	Maska
PE1-L0	203.0.113.40	/32
PE2-L0	203.0.113.41	/32
P3-L0	203.0.113.42	/32
P4-L0	203.0.113.43	/32

Síť zákazníka : **198.51.100.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S11 - VLAN10	32	198.51.100.0	198.51.100.1	198.51.100.30	198.51.100.1	/27
S12 - VLAN20	32	198.51.100.32	198.51.100.5	198.51.100.62	198.51.100.63	/27
S13	4	198.51.100.64	198.51.100.65	198.51.100.66	198.51.100.67	/30
S14	4	198.51.100.68	198.51.100.69	198.51.100.70	198.51.100.71	/30
S15	4	198.51.100.72	198.51.100.73	198.51.100.74	198.51.100.75	/30
S16	4	198.51.100.76	198.51.100.77	198.51.100.78	198.51.100.79	/30
S17	4	198.51.100.80	198.51.100.81	198.51.100.82	198.51.100.83	/30

Síť zákazníka : **192.0.2.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S18 - VLAN10	32	192.0.2.0	192.0.2.1	192.0.2.30	192.0.2.31	/27
S19 - VLAN20	32	192.0.2.32	192.0.2.33	192.0.2.62	192.0.2.63	/27
S20	4	192.0.2.64	192.0.2.65	192.0.2.66	192.0.2.67	/30
S21	4	192.0.2.68	192.0.2.69	192.0.2.70	192.0.2.71	/30
S22	4	192.0.2.72	192.0.2.73	192.0.2.74	192.0.2.75	/30
S23	4	192.0.2.76	192.0.2.77	192.0.2.78	192.0.2.79	/30
S24	4	192.0.2.80	192.0.2.81	192.0.2.82	192.0.2.83	/30

Obrázek 4: Adresní plán pro 1. scénář.

5.2 Scénář 2

Tento scénář je zaměřený na možnosti konfigurace L2 v jednotlivých simulačních prostředí. Směrování je zajištěno protokolem OSPF, který rozděluje navrženou topologii do oblastí. Páteřní oblast je označena jako Area0. Tato oblast propojuje zbylé dvě oblasti, označené jako Area10 a Area20. V oblastích Area10 a Area20 je nakonfigurován NAT a DHCP server, který přiděluje IP adresy do jednotlivých VLAN. Schéma navržené topologie je znázorněno na obrázku 6 a adresní plán pro Scénář 2 je čtenáři k dispozici na obrázku 7.

5.2.1 Realizace scénáře 2

Zajištění směrování

Jak již bylo uvedeno v úvodu o zajištění konektivity se stará protokol OSPF, který rozděluje navrhnutou topologii do oblastí. Nejdříve nakonfigurujeme IP adresy na rozhraní směrovačů podle adresního plánu a přiřadíme rozhraní směrovačů do požadovaných oblastí dle návrhu topologie.

```
interface serial 0/0
ip address 203.0.113.1 255.255.255.252
ip ospf 1 area 0

interface serial 0/1
ip address 203.0.113.14 255.255.255.252
ip ospf 1 area 10
```

Výpis 14: Ukázka konfigurace směrovacího protokolu OSPF rozděleného do oblastí na hraničním směrovači ABR1.

NAT overloading

Při použizí NAT overloadingu chceme překládat více adres z inside rozhraní směrovače na jednu adresu outside rozhraní směrovače s různými porty. Z tohoto důvodu nejdříve určíme vnitřní (inside) a vnější (outside) rozhraní směrovače. V dalším kroku vytvoříme access-list ve kterém určíme, které inside local adreasy se budou v NATu překládat. Jednotlivé access – listy pojmenujeme tak, aby jejich název odpovídal konkrétní VLAN. Nakonfigurovaný access – list odpovídá rozsahu adres pro danou VLAN. Na závěr již stačí povolit NAT overloading na vnějším rozhraní směrovače a přiřadit vytvořený access – list pro adresy, které se mají překládat.

```
interface Serial0/1
ip nat outside

interface FastEthernet1/0
ip nat inside
```

```
access-list 10 permit 198.51.100.0 0.0.0.31
ip nat inside source list 10 interface Serial0/1 overload
```

Výpis 15: Ukázka konfigurace NAT overloading pro VLAN10 v oblasti 10 na směrovači R5.

Default routa pro zajištění směrování z oblasti 10 přes NAT

```
ip route 0.0.0.0 0.0.0.0 Serial0/1
```

Výpis 16: Ukázka statické routy pro zajištění směrování z oblasti 10 přes NAT.

Vytvoření subinterface pro danou VLAN na směrovači pro spoj mezi směrovačem a přepínačem, na kterém je nakonfigurované trunk rozhraní. Default routa přes NAT

```
interface FastEthernet1/0.20
 encapsulation dot1Q 20
 ip address 198.51.100.33 255.255.255.224
```

Výpis 17: Vytvoření subinterface na rozhraní směrovače.

DHCP

Při konfiguraci DHCP nejdříve nakonfigurujeme adresní pool pro jednotlivé VLAN.

```
ip dhcp pool net10
network 198.51.100.0 255.255.255.224
default-router 198.51.100.1
exit

ip dhcp pool net20
network 198.51.100.32 255.255.255.224
default-router 198.51.100.33
exit
```

Výpis 18: Ukázka konfigurace poolu adres pro oblast 10.

Na závěr určíme od jaké IP adresy se mají adresy distribuovat jednotlivým zařízením.

```
ip dhcp excluded-address 198.51.100.0 198.51.100.1
ip dhcp excluded-address 198.51.100.32 198.51.100.33
```

Výpis 19: Vyloučení adres pro distribuci v oblasti 10.

Spanning Tree Protocol

Jak již bylo zmíněno v kapitole 4.2, simulační platforma GNS3 nepodporuje přepínání, tedy není možné použít Ethernet přepínač, který je standartně v nabídce zařízení v menu GNS3 a zapojit ho do smyčky. Navíc, tyto přepínače nemají konzolový přístup a je možné provádět pouze základní nastavení pomocí klikacího konfiguračního menu. Absence STP na Ethernet

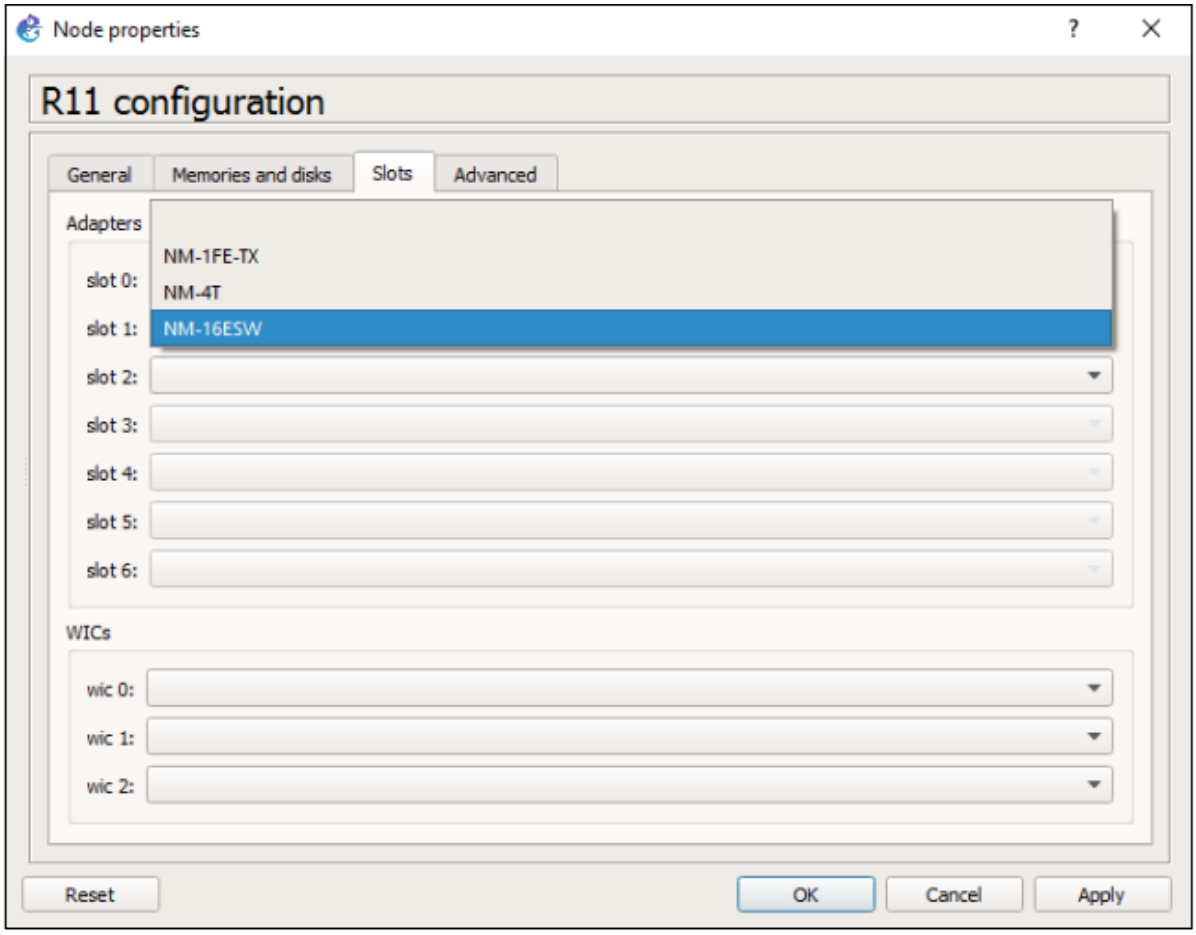
přepínači z nabídky GNS3 způsobí neovladatelnost GNS3 a uživatel je nucen vypnout GNS3 pomocí ukončení procesu a přijde o veškerou práci, kterou provedl po otevření projektu, ikdyž si svůj pokrok ukládal.

Řešení na absenci přepínání v GNS3 je použití cisco image, jako je c3725 a přepnout ho na přepínač, tedy vypnout směrování v konfigurační režimu. Tuto možnost jsem zvolil při realizaci tohoto scénáře.

```
no ip routing
```

Výpis 20: Vypnutí směrování.

V dalším kroku je nutné nastavit přepínací modul. Kliknutím na konkrétní zařízení zvolíme z nabídky položku V nastavení zvolíme položku Configure a poté záložku Slots. Z nabídky vybereme NM – 16ESW (16-port, 10/100 Ether – Switch Network Module). Díky použití tohoto modulu můžeme konfigurovat jednotlivé VLANy a je spuštěn STP.



Obrázek 5: Nastavení přepínacího modulu.

Nastavení portů na směrovači pro účely přepínání do módu switchport.

```
interface range Fa1/0 - 15
switchport
```

Výpis 21: Nastavení portu jako switchport.

Vytvoření a pojmenování VLANy.

```
vlan 10
name S6
exit
```

Výpis 22: Vytvoření VLANy.

Nastavení portů jako trunk a přidání VLAN 10 a 20 do trunku.

```
interface range Fa1/0 - 2
switchport mode trunk
switchport trunk allowed vlan add 10,20
```

Výpis 23: Konfigurace rozhraní trunk.

Konfigurace přístupových portů pro jednotlivé PC.

```
interface f1/3
switchport mode access
switchport access vlan 10
```

Výpis 24: Konfigurace přístupových portů.

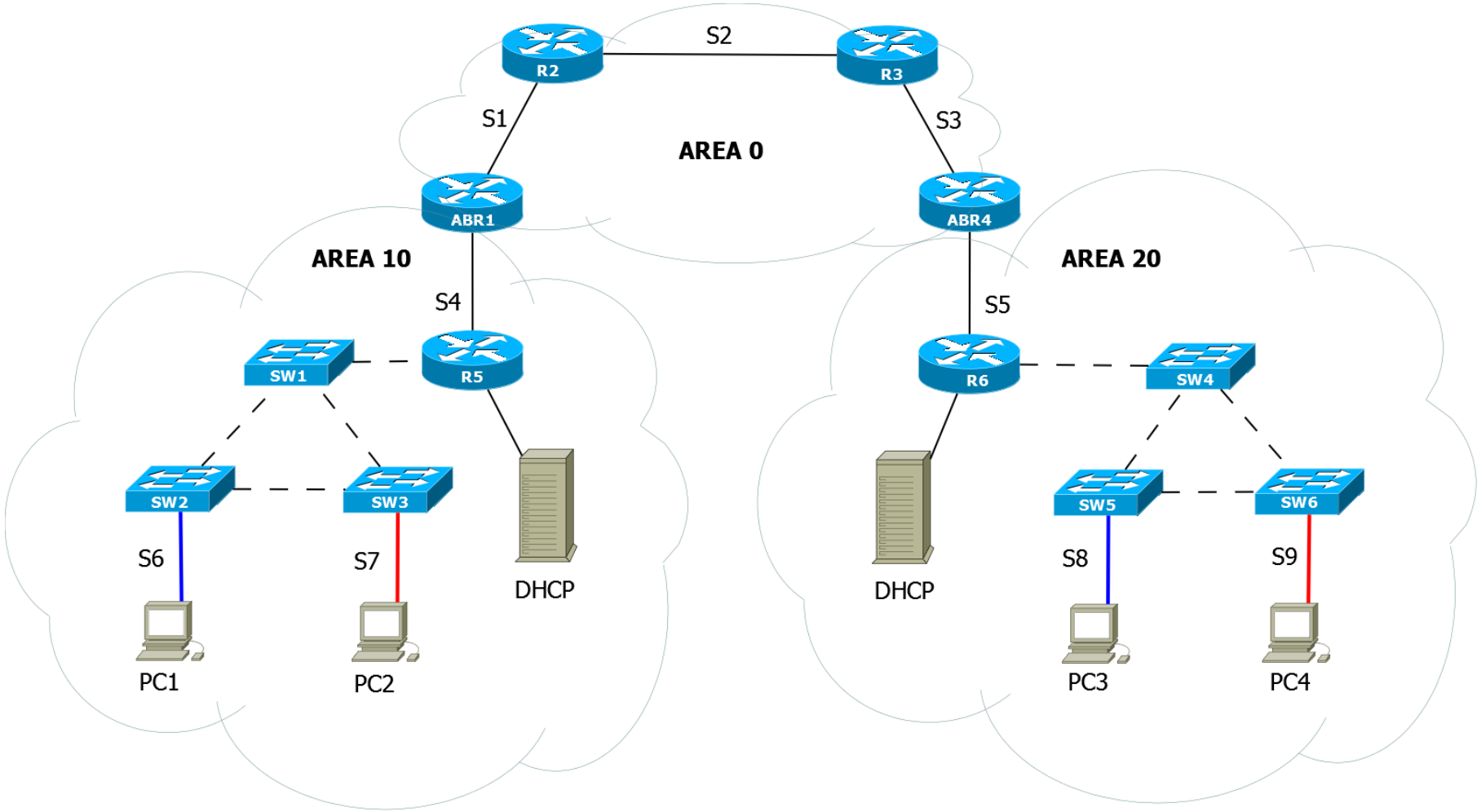
Aby bylo možné umožnit portu na kterém je připojena koncová stanice přejít po zapnutí ihned do stavu forwarding, je nutné nastavit takový port jako PortFast.

```
interface f1/3
spanning-tree portfast
```

Výpis 25: Konfigurace PortFast.

Zde je nutné zdůraznit, že prostředí VIRL podporuje technologie L2 a proto stačí použít zařízení IOSvL2, které umožňuje uživateli provádět konfigurace jako u běžných přepínačů od firmy Cisco a nemusí se uživatel spokojit s PVST jako je tomu v případě simulační platformy GNS3.

Kompletní výpisy konfigurací jednotlivých zařízení jsou k dispozici v příloze v adresáři Konfigurace/scenar2.



Obrázek 6: Schéma scénáře 2.

Síť poskytovatele: **203.0.113.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S1	4	203.0.113.0	203.0.113.1	203.0.113.2	203.0.113.3	/30
S2	4	203.0.113.4	203.0.113.5	203.0.113.6	203.0.113.7	/30
S3	4	203.0.113.8	203.0.113.9	203.0.113.10	203.0.113.11	/30
S4	4	203.0.113.12	203.0.113.13	203.0.113.14	203.0.113.15	/30
S5	4	203.0.113.16	203.0.113.17	203.0.113.18	203.0.113.19	/30

Loopback rozhraní	Adresa	Maska
ABR1-L0	203.0.113.24	/32
R2-L0	203.0.113.25	/32
R3-L0	203.0.113.26	/32
ARR4-L0	203.0.113.27	/32
R5-L0	203.0.113.28	/32
R6-L0	203.0.113.29	/32

Síť zákazníka: **198.51.100.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S6- VLAN10	32	198.51.100.0	198.51.100.1	198.51.100.30	198.51.100.31	/27
S7- VLAN20	32	198.51.100.32	198.51.100.33	198.51.100.62	198.51.100.63	/27

Síť zákazníka: **192.0.2.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S8- VLAN10	32	192.0.2.0	192.0.2.1	192.0.2.30	198.0.2.31	/27
S9- VLAN20	32	192.0.2.32	192.0.2.33	192.0.2.62	198.0.2.63	/27

Obrázek 7: Adresní plán pro 2. scénář.

5.3 Scénář 3

Poslední scénář byl zaměřen na technologie virtuálních privátních sítí (VPN).

V páteřní síti poskytovale je provozován link – state směrovací protokol OSPF s technologií MPLS, která pro výměnu značek uvnitř páteřní sítě poskytovatele využívá standartizovaný protokol LDP.

Propojení síťové L3 vrstvy modelu OSI, bylo realizováno pomocí technologie MPLS – VPN a pro propojení L2 vrstvy modelu OSI byla použita technologie AToM.

Schéma navržené topologie je znázorněno na obrázku 8 a adresní plán pro Scénář 3 je čtenáři k dispozici na obrázku 9.

5.3.1 Realizace scénáře 3

Směrování a aktivace MPLS v páteřní síti poskytovatele

Nejdříve na všech směrovačích poskytovatele nakonfigurujeme Loopback rozhraní a IP adresy na rozhraních směrovačů, které jsou připojeny do páteřní sítě poskytovatele.

V dalším kroku spustíme protokol OSPF. Do oblasti 0 vypropagujeme jednotlivá rozhraní směrovačů poskytovatele (včetně Loopback rozhraní), které jsou přímo připojené do páteřní sítě poskytovatele. Dále na rozhraních směrovačů poskytovatele (kromě Loopback rozhraní) aktivujeme technologii MPLS.

```
PE1:
   router ospf 1

   interface loopback 0
      ip address 203.0.113.33 255.255.255.255
      ip ospf 1 area 0
      exit

   interface serial 1/0
      description PE1-P-N1
      ip address 203.0.113.2 255.255.255.252
      mpls ip
      ip ospf 1 area 0
      no shutdown
      exit
```

Výpis 26: Ukázka konfigurace rozhraní směrovače PE1.

Pro potřeby distribuce značek pomocí LDP použijeme jednotlivé Loopback rozhraní jako identifikátory.

```
P-PE4:
   mpls label protocol ldp
   mpls ldp router-id Loopback0 force
```

AToM

Mezi hraničními směrovači poskytovatele (PE1, PE3) vytvoříme virtuální L2 kanál, který je zapouzdřen v technologii MPLS. Jako I=identifikátor virtuálního kanálu zvolíme číslo 10, které v našem případě reprezentuje VLAN 10.

```
PE1:
   interface fastEthernet 0/0
      xconnect 203.0.113.35 10 encapsulation mpls
      no shutdown

PE3:
   interface fastEthernet 0/0
      xconnect 203.0.113.33 10 encapsulation mpls
      no shutdown
```

Výpis 27: Ukázka vytvoření virtuálního L2 kanálu mezi směrovači PE1 a PE2.

MPLS – VPN

V našem scénáři chceme zprovoznit MPLS-VPN mezi autonomními systémy AS110 – AS120 a AS130 – AS140. Nejdříve na hraničních směrovačích poskytovatele PE2 a PE4 vytvoříme virtuální směrovací tabulky pro jednotlivé zákazníky, v našem případě VRFA a VRFB. Pro odlišení přenášených síťových prefixů přiřadíme každé VRF route distinguisher (RD) a definujeme, jaké cesty chceme do dané VRF importovat.

```
PE4:
   ip vrf VRFA
      rd 110:1
      route-target export 110:1
      route-target import 120:1
```

Výpis 28: Ukázka konfigurace vytvoření virtuální směrovací tabulky pro VRFA na hraničním směrovači poskytovatele PE4.

V dalším kroku přiřadíme vytvořené VRF na rozhraní hraničních směrovačů poskytovatele, které jsou připojeny k směrovačům zákazníka a na tyto rozhraní přiřadíme IP adresy.

```
PE4:
   interface Serial0/0
      ip vrf forwarding VRFA
      ip address 198.51.100.141 255.255.255.252
      no shutdown
      exit

CE1:
   interface serial 0/0
      ip address 198.51.100.142 255.255.255.252
      no shutdown
      exit
```

Výpis 29: Ukázka přiřazení IP adres na rozhraní směrovačů PE4 a CE1. Dále přiřazení rozhraní hraničního směrovače PE4 do VRFA.

Pro účely přenášení VPNv4 prefixů (RD a prefix IPv4) uvnitř AS100 poskytovatele mezi vzdálenými směrovači zákazníka využijeme protokol MP – BGP (Multiprotocol BGP). Pro tyto účely navážeme spojení mezi Loopback rozhraními hraničních směrovačů PE2 a PE4.

```
PE4:
   router bgp 100
      neighbor 203.0.113.34 remote-as 100
      neighbor 203.0.113.34 update-source Loopback0
      no auto-summary

      address-family vpnv4
         neighbor 203.0.113.34 activate
         neighbor 203.0.113.34 send-community both
         exit-address-family
```

Výpis 30: Ukázka konfigurace MP – BGP pro přenášení VPNv4 prefixů unitř páteřní sítě poskytovatele na hraničním směrovači PE4.

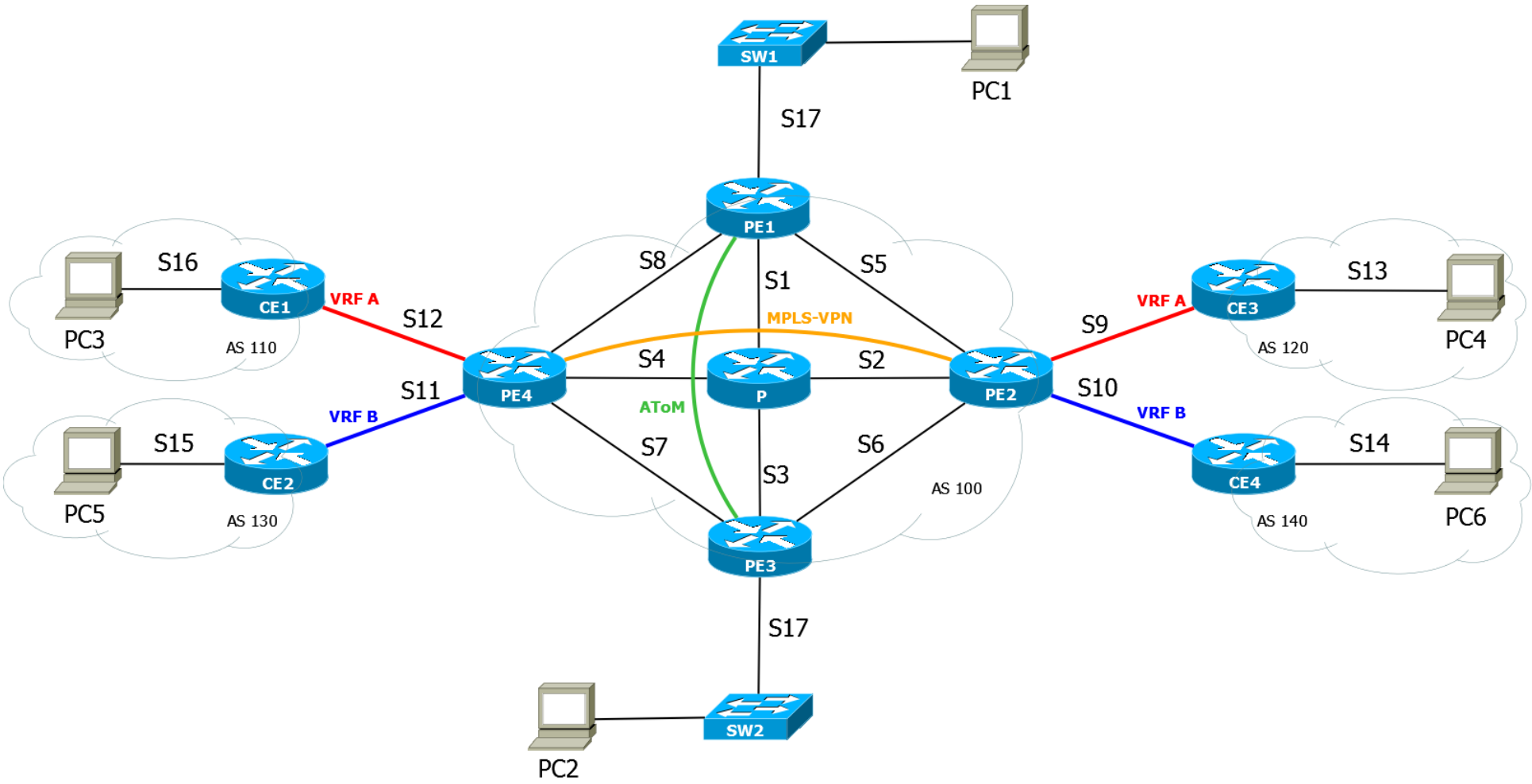
Jako poslední krok zbývá konfigurace směrování mezi směrovači poskytovatele (PE) a zákazníka (CE).

```
PE4:
   address-family ipv4 vrf VRFA
      neighbor 198.51.100.142 remote-as 110
      neighbor 198.51.100.142 activate
      no synchronization
      exit-address-family

CE1:
   router bgp 110
      neighbor 198.51.100.141 remote-as 100
      redistribute connected
```

Výpis 31: Ukázka konfigurace mezi směrovači poskytovatele a zákazníka pro VRFA.

Kompletní výpisy konfigurací jednotlivých zařízení jsou k dispozici v příloze v adresáři Konfigurace/scenar3.



Obrázek 8: Schéma scénáře 3.

Síť poskytovatele : **203.0.113.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S1	4	203.0.113.0	203.0.113.1	203.0.113.2	203.0.113.3	/30
S2	4	203.0.113.4	203.0.113.5	203.0.113.6	203.0.113.7	/30
S3	4	203.0.113.8	203.0.113.9	203.0.113.10	203.0.113.11	/30
S4	4	203.0.113.12	203.0.113.13	203.0.113.14	203.0.113.15	/30
S5	4	203.0.113.16	203.0.113.17	203.0.113.18	203.0.113.19	/30
S6	4	203.0.113.20	203.0.113.21	203.0.113.22	203.0.113.23	/30
S7	4	203.0.113.24	203.0.113.25	203.0.113.26	203.0.113.27	/30
S8	4	203.0.113.28	203.0.113.29	203.0.113.30	203.0.113.31	/30

Loopback rohraní	Adresa	Maska
P-L0	203.0.113.32	/32
PE1-L0	203.0.113.33	/32
PE2-L0	203.0.113.34	/32
PE3-L0	203.0.113.35	/32
PE4-L0	203.0.113.36	/32

Síť zákazníka: **198.51.100.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S13	32	198.51.100.0	198.51.100.1	198.51.100.30	198.51.100.31	/27
S14	32	198.51.100.32	198.51.100.33	198.51.100.62	198.51.100.63	/27
S15	32	198.51.100.64	198.51.100.65	198.51.100.94	198.51.100.95	/27
S16	32	198.51.100.96	198.51.100.97	198.51.100.126	198.51.100.127	/27
S9	4	198.51.100.128	198.51.100.129	198.51.100.130	198.51.100.131	/30
S10	4	198.51.100.132	198.51.100.133	198.51.100.134	198.51.100.135	/30
S11	4	198.51.100.136	198.51.100.137	198.51.100.138	198.51.100.139	/30
S12	4	198.51.100.140	198.51.100.141	198.51.100.142	198.51.100.143	/30

Síť zákazníka: **192.0.2.0 /24**

Název sítě	Počet adres	Prefix	První dostupná adresa	Poslední dostupná adresa	Broadcast	Maska
S17	255	192.0.2.0	192.0.2.1	192.0.2.254	192.0.2.255	/24

Obrázek 9: Adresní plán pro 3. scénář.

6 Zachycení provozu

Simulační platforma GNS3 umožňuje zachytávání provozu prostřednictvím síťového analyzátoru Wireshark. Wireshark je podobný programu tcpdump s rozdílem, že Wireshark obsahuje daleko více analyzátorů telekomunikačního provozu a disponuje grafickým uživatelským rozhraním, díky kterému se Wireshark stává velice snadno a pohodlně ovladatelným. Wireshark je pravděpodobně jeden z nejlepších open source paketových analyzátorů současnosti.

V prostředí VIRL je možné zachytávat provoz do formátu podporovaným paketovým analyzátorem Wireshark. Wireshark používá pro zachytávání provozu souborový formát libpcap, který je možné otevřít i v jiných aplikacích, které používájí stejný souborový formát tedy např. tcpdump.

6.1 GNS3

Nastavení zachytávání je možné prostřednictím nabídky GNS3 po kliknutí na síťový prvek navrhnuté topologie. Po zvolení Capture je uživatel vyzván, aby zvolil port síťového prvku, na kterém se má spustit odchytávání provozu a otevře se program Wireshark. Po ukončení zachytávání provozu si může uživatel uložit získané hodnoty ve formátu .pcapng (PCAP Next Generation Dump File Format).

Zachytátávání provozu prostřednictvím Wiresharku je možné na portech zvolených síťových prvků. GNS3 neumožňuje zachytávání provozu na koncových stanicích ať už se jedná o VPCS, a nebo virtualizované koncové stanice z VirtualBoxu. Pro zachytávání provozu na koncových virtualizovaných stanicích je nutné spustit paketový analyzátor přímo uvnitř koncové stanice, např. program tcpdump.

6.2 VIRL

V prostředí VIRL existují dvě možnosti jak nastavit zachytávání provozu. První možnost je přímo na serveru, na kterém je nainstalovaný VIRL prostřednictvím webového rozhraní.
Po přihlášení do VIRLu prostřednictvím webového rozhraní je nutné dodržet následující postup:

1. V položce Project simulations zvolit simulaci, ve které požadujeme zachytávání provozu.
2. V části pojmenované Interfaces je seznam všech dostupných rozhraních naší simulace (i koncových stanic). Zvolíme požadované rozhraní, na kterých si přejeme zachytávat provoz a potvrdíme tlačítkem Traffic Capture.
3. Uživateli je zobrazen formulář Create Capture. Zde můžeme nastavit PCAP filtr, limity zachytávání provozu (počet paketů, velikost a čas zachytávání) a mód zachytávání (offline, live).

Druhá možnost je přímo v prostředí VM Maestro. Zachytávání provozu je podobné jako v GNS3. Stačí zvolit požadované rozhraní, vyplnit formulář Create Capture(totožný jako na webu) a spustit zachytávání provozu.

V obou případech po vytvoření zachytávání je spustěn proces zachytávání provozu a po ukončení zachytávání je možné stáhnout zachycený provoz ve formátu .pcap.

6.3 Testování zachycení provozu

Pro testování zachycení provozu jsem zvolil scénář číslo 3 a cílem bylo zachytit ICMP zprávy z koncové stanice PC3 v AS110 na koncovou stanici PC4 umístěnou v AS120. K tomuto účelu byl použit nástroj ping s parametrem -c, který udává kolikrát se má vygenerovat Echo Request. V našem případě byl zvolen parametr -c roven 5. V tomto experimentu šlo o porování získaných hodnot ze zachycení provozu pomocí programu Wireshark a tcpdump.

Jelikož prostředí GNS3 neumožňuje zachytávání provozu na portech koncových stanic, ale je nutné spustit paketový analyzátor přímo uvnitř koncové stanice, bylo nutné na rozhraní koncových stanic (virtualizovaný OS Lubuntu) spustit tcpdump. Program tcpdump byl spuštěn aby zaznamenával pouze icmp protokol a naslouchal na rozhraní enp0s3. Dále bylo nastaveno umístění souboru, do kterého se zaznamenává zachycený provoz. Na rozhraní Serial0/3 směrovače P byl spuštěn Wireshark.

```
tcpdump -i enp0s3 icmp -w /home/vec0027/Desktop/pc3.pcap
```

Výpis 32: Spuštění zachytávání ICMP zpárv na rozhraní enp0s3 v OS Lubuntu.

Zachycený provoz prostřednictvím programu Wireshark a tcpdump z prostředí GNS3 je dostupný v příloze, konkrétně v adresáři Zachycení provozu/GNS3. Jedná se o soubory *pc*3.*pcap*, $router_P$.*pcap* a *pc*4.*pcap*.

V prostředí VIRL byl proveden totožný test s rozdílem, že k zachytávání provozu nebyl použit tcpdump a konfigurace zachytávání provozu byla provedena prostřednictvím webového VIRL.

Takto zachycený provoz je součást přílohy a nachází se v adresáři Zachycení provozu/VIRL a jedná se o soubory *pc*3.*pcap*, *P.pcap* a *pc*4.*pcap*. Získané soubory zachycující ICMP zprávy z prostředí GNS3 a VIRL se od sebe nijak neliší.

7 Testování scénářů

Pro testování navrhnuté topologie jsem zvolil volně dostupný multiplatformní nástroj iPerf. Nástroj iPerf umožňuje měřit maximální dosažitelnou přenosovou rychlost v IP sítích. Dále je možné pro jednotlivé testy nastavit parametry týkající se načasování a použitých protokolů UDP a TCP.

Simulační platforma GNS3 používá pro simulaci hostů tzv. Virtuální PC (VPCs). Tyto virtuální PC se nehodí pro moje účely testování, jelikož na nich lze kromě nastavení IP adresy generovat pouze ping a není možné do nich cokoliv nainstalovat. Z toho důvodu jsem pro účely testování použil dvě instance virtualizovaných PC na platformě VirtualBox. Operační systém jsem zvolil Lubuntu 16.04 z důvodu nízkých požadavků na výkon a do nich nainstaloval nástroj iPerf. Jedna instance v testovacím scénáři vystupuje v serverovém režimu a druhá klientském režimu.

Veškeré testy zaměřené na simulační platformu GNS3 byly testovány na Operačním systému Microsoft Windows 10 Home 64bit (verze 1607, build operačního systému 14393.1066) i Linuxu Ubuntu 16.04 LTS Xenial Xerus 64bit (verze kernelu: 4.4.0-72-generic).

Simulační platforma VIRL poskytuje několik uzlů, které je možné použít jako koncové zařízení. Pro moje testovací účely jsem zvolil uzel lxc – iperf, což je linuxový kontejner, který obsahuje nástroj iPerf. Způsob práce s nástrojem iPerf v linuxovém kontejneru Lxc-iperf se nijak neliší od standartních OS.

Použitý hardware pro GNS3
CPU: Intel(R) Core(TM) i7-3630QM CPU@ 2.40GHz
RAM: 8,00GB (použitelné 7,86 GB)

Použitý hardware pro VIRL
CPU: Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz
RAM: 32GB

Profesionální řešení VIRL nebylo možné použít na stejném stroji jako GNS3 kvůli licenčním podmínkám, ale jak je vidět v tabulce 2, oba procesory jsou si podle PassMark score výkonem zhruba srovnatelné.

Porovnání CPU [10]

Tabulka 2: Porovnání CPU.

	Intel Xeon E3-1220 @ 3.10GHz	Intel Core i7-3630QM @ 2.40GHz
Class CPU	Server	Laptop
Clockspeed	3.1 GHz	2.4 GHz
Turbo Speed	až 3,4 GHz	až 3,4 GHz
Sigle Thread Rating	1718	1685
CPU Mark	6053	7595

Testování UDP

Na klientské stanici postupně měníme nastavenou šířku pásma a pozorujeme naměřenou hodnotu přenosové rychlosti, jitter[8] a ztrátovost. UDP datagramy generujeme v intervalu 1 sekunda po dobu 30 sekund.

```
iperf -s -u
```

Výpis 33: Spuštění serveru pro příjem UDP datagramů.

```
iperf -c Adresa serveru -i interval(s) -b nastavena sirka pasma(Mb) -t celkova
    doba testu(s)
```

Výpis 34: Ukázka nastavení klientské stanice pro generování UDP datagramů.

Testování TCP

Klientská stanice naváže spojení se serverem a začne maximální možnou rychlostí posílat data na server. Stanice následně po ukončení přenosu zobrazí informace o přenesých datech a šířce pásma na straně serveru a na straně klienta.

```
iperf -s
```

Výpis 35: Spuštění serveru pro příjem TCP spojení.

```
iperf -c Adresa serveru
```

Výpis 36: Spuštění klienta pro generování TCP spojení.

Testování odezvy pomocí nástroje PING

Klientská stanice posílá IP datagramy na server. V tomto testu nás zajímá délka zpoždění odpovědi.

[8] V sítích IP se jedná o změnu latence toku paketů mezi dvěma systémy. Tento jev je často způsoben přetížením sítě, nebo změnou trasy.

7.1 Scénář 1

Test 1.1

Test zaměřený na propustnost z AS110. Klientská stanice PC1 generuje provoz na server, který je umístěn místo směrovače poskytovatele PE1.

Testování přenosové rychlosti pomocí UDP

Tabulka 3: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Windows pro Test 1.1.

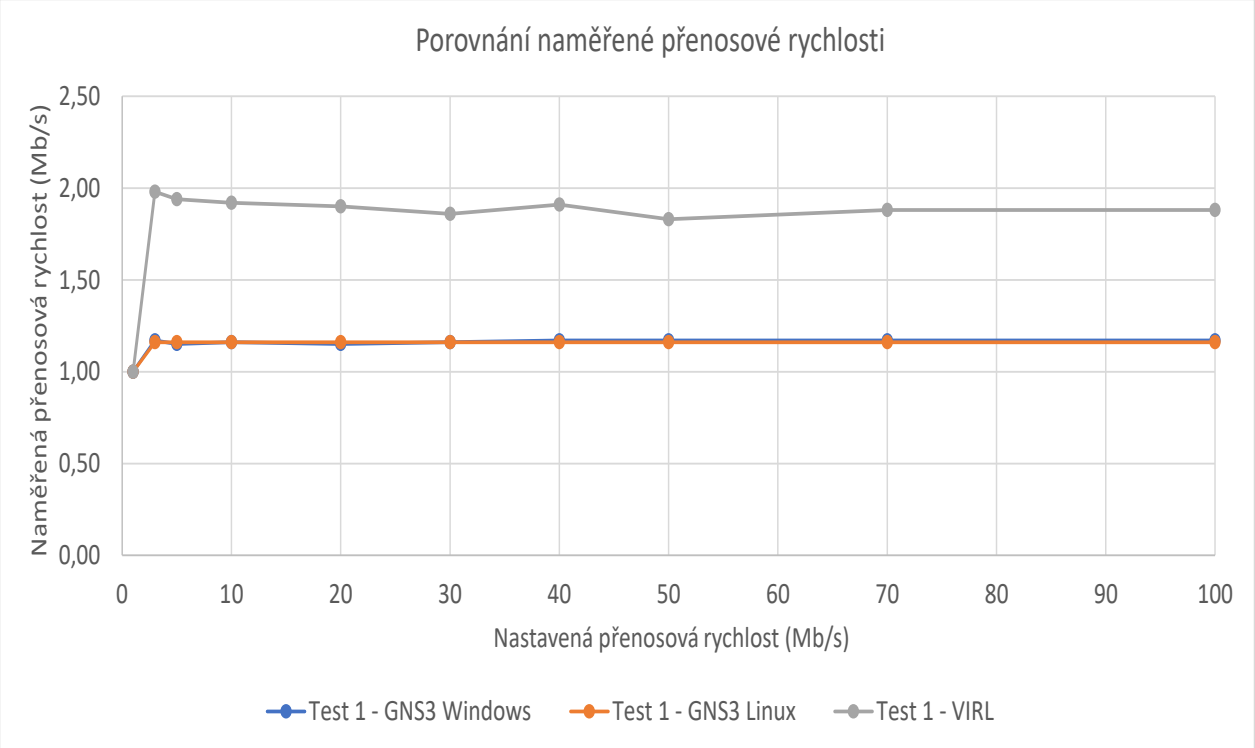
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	0,98	2,780	0/ 2552 (0%)
3	0,0-31,0	4,34	1,17	4,241	4548/ 7644 (59%)
5	0,0-31,2	4,27	1,15	3,452	9574/12619 (76%)
10	0,0-31,2	4,32	1,16	4,110	21287/24372 (87%)
20	0,0-31,2	4,29	1,15	2,495	25793/28851 (89%)
30	0,0-31,3	4,32	1,16	2,615	25111/28192 (89%)
40	0,0-31,1	4,33	1,17	3,481	28913/32003 (90%)
50	0,0-31,1	4,34	1,17	3,423	29275/32369 (90%)
70	0,0-31,0	4,34	1,17	3,338	29083/32177 (90%)
100	0,0-31,0	4,33	1,17	2,960	28150/31239 (90%)

Tabulka 4: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Linux pro Test 1.1.

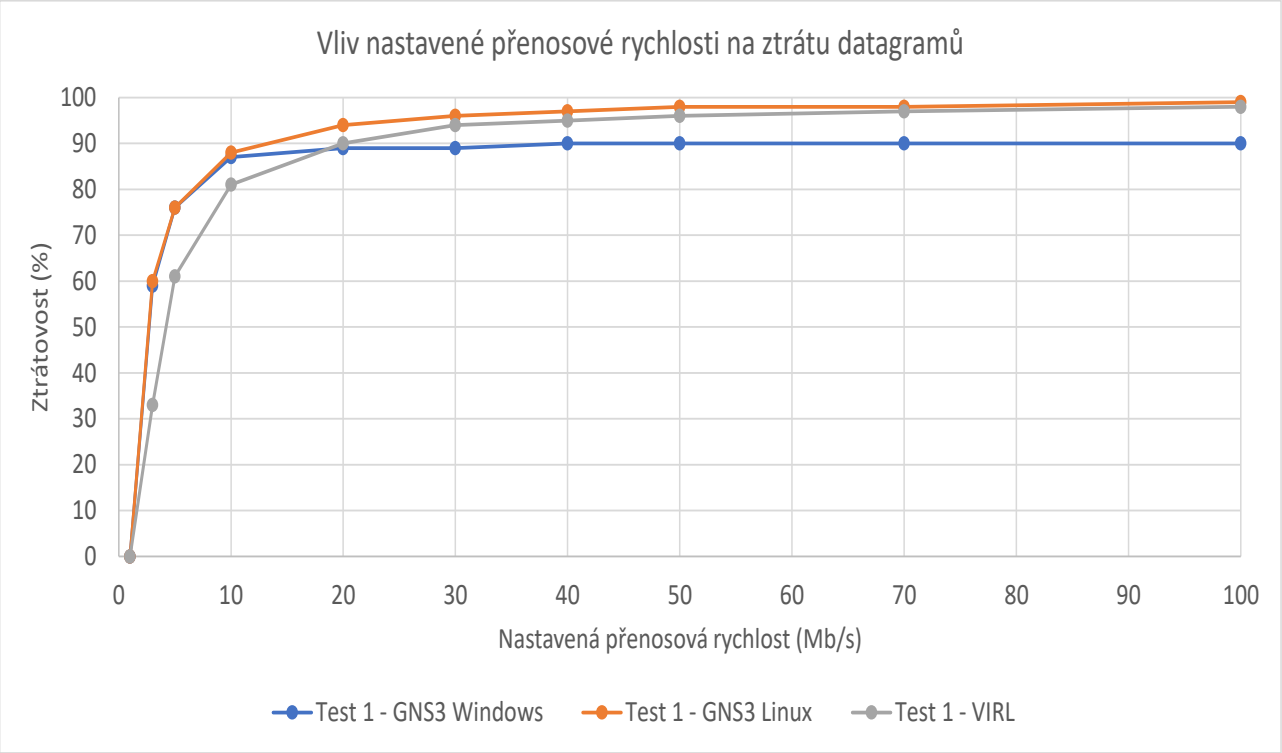
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	2,798	0/ 2552 (0%)
3	0,0-31,1	4,31	1,16	3,934	4583/ 7654 (60%)
5	0,0-31,1	4,31	1,16	4,671	9681/12756 (76%)
10	0,0-31,1	4,31	1,16	2,888	22436/25512 (88%)
20	0,0-31,2	4,31	1,16	3,340	47946/51018 (94%)
30	0,0-31,0	4,31	1,16	3,771	73454/76525 (96%)
40	0,0-31,1	4,31	1,16	3,212	98934/102008 (97%)
50	0,0-31,1	4,31	1,16	3,451	124582/127654 (98%)
70	0,0-31,1	4,31	1,16	3,281	175420/178493 (98%)
100	0,0-31,1	4,30	1,16	3,091	252703/255771 (99%)

Tabulka 5: Naměřené hodnoty pro UDP datagramy v prostředí VIRL pro Test 1.1.

Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	1,608	0/ 2552 (0%)
3	0,0-30,3	7,14	1,98	16,279	2563/ 7655 (33%)
5	0,0-30,5	7,05	1,94	31,877	7728/12756 (61%)
10	0,0-30,2	6,92	1,92	16,246	20575/25509 (81%)
20	0,0-30,3	6,87	1,90	16,639	46122/51020 (90%)
30	0,0-30,2	6,71	1,86	18,250	71637/76425 (94%)
40	0,0-30,2	6,89	1,91	16,245	97078/101991 (95%)
50	0,0-30,2	6,60	1,83	17,772	122949/127654 (96%)
70	0,0-30,7	6,90	1,88	8,400	173608/178527 (97%)
100	0,0-30,7	7,96	1,88	3,396	250446/256127 (98%)



Obrázek 10: Porovnání naměřených přenosových rychlostí pro test 1.1 ve scénáři 1.



Obrázek 11: Zobrazené vlivu nastavené přenosové rychlosti na ztrátu datagramů pro test 1.1 ve scénáři 1.

Test TCP spojení

Tabulka 6: Test 1.1 pro TCP spojení ve scénáři 1.

Prostředí	Přeneseno (MB)	Přenosová rychlost Klient (Mb/s)	Přenosová rychlost Server (Mb/s)
GNS3 – Windows	2,25	1,66	1,16
GNS3 – Linux	2,12	1,62	1,14
VIRL	3,05	2,41	2,1

Testování odezvy pomocí nástroje PING

Tabulka 7: Testování odezvy pro test 1.1 ve scénáři 1.

Prostředí	Min(ms)	Průměr(ms)	Max(ms)	Odchylka(ms)	Ztráta(%)
GNS3 – Windows	71,79	86,94	108,11	9,74	0
GNS3 – Linux	61,24	73,27	106,02	10,67	0
VIRL	3,99	7,42	56,20	8,21	0

Vyhodnocení testu 1.1:

Z hodnot naměřených během testu přenosové rychlosti můžeme určit z grafu na obrázku 10 , že nejvyšší přenosové rychlosti bylo dosaženo v prostředí VIRL. Naměřená přenosová rychlost konvergovala k hodnotě 1,9Mb/s. Zaměříme – li se na ztrátu datagramů během testování, potom dle grafu na obrázku 11 nejmenší ztrátovost dosáhlo prostředí GNS3 na OS Windows, kde hodnota ztráty datagramů konverguje u vyšších přenosových rychlostí k 90%. Naměřený jitter dosahoval nejvyšších hodnot v prostředí VIRL, viz. tabulka 5.

Zajímavé může být pozorování časů jednotlivých intervalů přenosu, i když byl nastaven interval na 30 sekund, přenos ve vetšině případů trval o něco déle. Tento jev je způsoben tím, že iPerf klient komunikuje s iPerf serverem , který má vyrovnávací paměť. Klient se domnívá, že celý přenos byl již proveden ještě před tím, než server skutečně obdržel poslední blok dat. Tento jev je možné pozorovat ve většině provedených testů.

Z testu TCP spojení můžeme říct, dle tabulky 6, že nejvyšší přenosové rychlosti bylo dosaženo v prostředí VIRL, z čehož i plyne nejvíce přenesených dat. Bylo přeneseno 3,05MB dat a naměřená přenosová rychlost na straně klienta dosáhla hodnoty 2,41Mb/s a na straně serveru 2,1Mb/s.

V případě testování doby odezvy, jasně dominuje prostředí VIRL, které dle tabulky 7 dosahuje v průměru 7,42ms.

Test 1.2

Test zaměřený na propustnost celkového navrhnutého řešení. Klientská stanice umístěna v AS110 generuje provoz přes AS100 poskytovalte do AS120 na stanici PC3.

Testování přenosové rychlosti pomocí UDP

Tabulka 8: Naměřené hodnoty během generování UDP datagramů v prostředí GNS3 na platformě Windows pro Test 1.2.

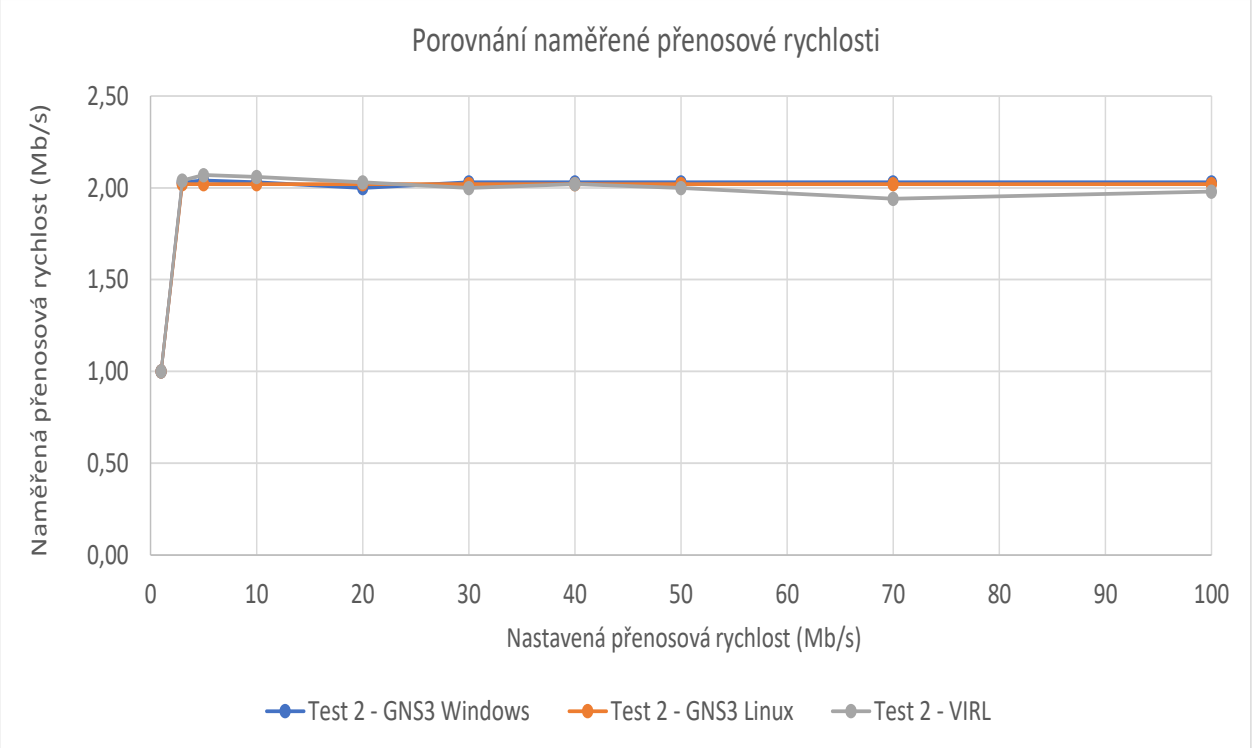
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	0,98	3,012	0/ 2552 (0%)
3	0,0-30,4	7,38	2,03	5,409	2389/ 7650 (31%)
5	0,0-30,4	7,37	2,04	9,437	7481/12738 (59%)
10	0,0-30,4	7,37	2,03	4,634	19748/25007 (79%)
20	0,0-30,4	7,26	2,00	3,108	26967/32146 (84%)
30	0,0-30,4	7,37	2,03	8,648	32933/38190 (86%)
40	0,0-30,4	7,37	2,03	6,179	31046/36304 (86%)
50	0,0-30,4	7,37	2,03	6,152	33202/38461 (86%)
70	0,0-30,4	7,37	2,03	4,115	33424/38684 (86%)
100	0,0-30,4	7,37	2,03	6,319	33337/38595 (86%)

Tabulka 9: Naměřené hodnoty během generování UDP datagramů v prostředí GNS3 na platformě Linux pro Test 1.2.

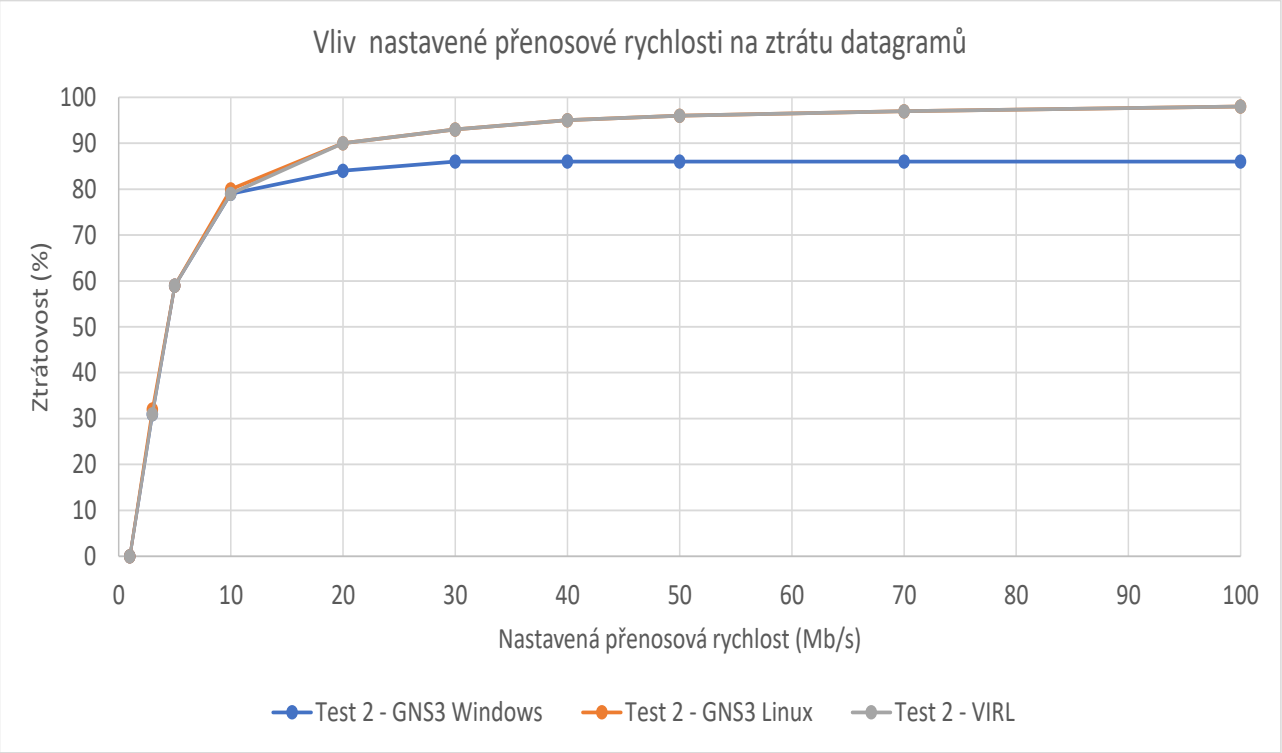
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	0,98	4,609	0/ 2552 (0%)
3	0,0-30,4	7,33	2,02	5,869	2426/ 7654 (32%)
5	0,0-30,4	7,33	2,02	5,443	7529/12756 (59%)
10	0,0-30,4	7,33	2,02	4,331	20282/25510 (80%)
20	0,0-30,4	7,33	2,02	6,082	45789/51015 (90%)
30	0,0-30,4	7,33	2,02	7,003	71290/76518 (93%)
40	0,0-30,4	7,33	2,02	5,002	96800/102028 (95%)
50	0,0-30,4	7,33	2,02	7,355	122429/127656 (96%)
70	0,0-30,4	7,32	2,02	5,529	173318/178542 (97%)
100	0,0-30,4	7,33	2,02	5,491	251020/256246 (98%)

Tabulka 10: Naměřené hodnoty během generování UDP datagramů v prostředí VIRL pro Test 1.2.

Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	3,759	0/ 2553 (0%)
3	0,0-30,3	7,37	2,04	16,336	2395/ 7654 (31%)
5	0,0-30,0	7,39	2,07	0,219	7487/12756 (59%)
10	0,0-30,0	7,35	2,06	1,076	20263/25507 (79%)
20	0,0-30,2	7,30	2,03	16,418	45812/51020 (90%)
30	0,0-30,2	7,22	2,00	16,067	71352/76503 (93%)
40	0,0-30,0	7,22	2,02	0,248	96694/101843 (95%)
50	0,0-30,2	7,20	2,00	16,027	122516/127653 (96%)
70	0,0-30,5	7,05	1,94	31,949	172962/177993 (97%)
100	0,0-33,7	7,96	1,98	3,396	250446/256127 (98%)



Obrázek 12: Porovnání naměřených přenosových rychlostí pro test 1.2 ve scénáři 1.



Obrázek 13: Zobrazené vlivu nastavené přenosové rychlosti na ztrátu datagramů pro test 1.2 ve scénáři 1.

Test TCP spojení

Tabulka 11: Test 1.2 pro TCP spojení ve scénáři 1

Prostředí	Přeneseno (MB)	Přenosová rychlost Klient (Mb/s)	Přenosová rychlost Server (Mb/s)
GNS3 – Windows	3,12	2,42	2,01
GNS3 – Linux	3,12	2,41	1,99
VIRL	2,7	2,14	1,91

Testování odezvy pomocí nástroje PING

Tabulka 12: Testování odezvy pro test 1.2 ve scénáři 1

Prostředí	Min(ms)	Průměr(ms)	Max(ms)	Odchylka(ms)	Ztráta(%)
GNS3 – Windows	21,28	28,08	37,61	4,84	0
GNS3 – Linux	20,529	25,779	36,579	3,376	0
VIRL	0,98	4,36	23,22	4,36	0

Vyhodnocení testu 1.2:

Z grafu na obrázku 12 můžeme určit, že měření přenosové rychlosti dopadlo velmi podobně v obou simulačních prostředí. Přenosová rychlost konvergovala k hodnotě 2Mb/s. Co se týče ztráty datagramů, tak z grafu na obrázku 13 je viditelné, že simulační prostředí GNS3 na OS Windows dosahovalo během testování nejmenší ztrátu datagramů. Hodnota ztrátovosti konverguje u vyšších přenosových rychlostí k 86%. Při pozorování naměřených hodnot jitteru je vidět z tabulky 10, že nejhorších hodnot dosáhlo prostředí VIRL.

Dle tabulky 11 můžeme říct, že přenosová rychlost během testování TCP spojení byla nejvyšší v prostředí GNS3. Naměřená hodnota přenosové rychlosti na straně klienta se pohybovala kolem hodnoty 2Mb/s a na straně serveru kolem hodnoty 2,4Mb/s. Dále se podařilo přenést celkem 3,12MB dat. Tyto hodnoty jsou shodné při použití GNS3 v OS Windows i Linux.

Naměřené hodnoty v tabulce 12 ukazují, že nejlepších výsledků při testování odezvy bylo dosaženo v prostředí VIRL. Průměrná doba odezvy dosahuje hodnoty 4,36ms.

7.2 Scénář 2

Test 2.1

Test zaměřený na propustnost v oblasti 10 přes NAT, který je nakonfigurovaný na směrovači R5. Klientská stanice připojená do přepínače S6 umístěná v oblasti 10 generuje provoz na server, který je umístěn za směrovačem R5, tedy nahrazuje směrovač ABR1.

Testování přenosové rychlosti pomocí UDP

Tabulka 13: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Windows pro Test 2.1.

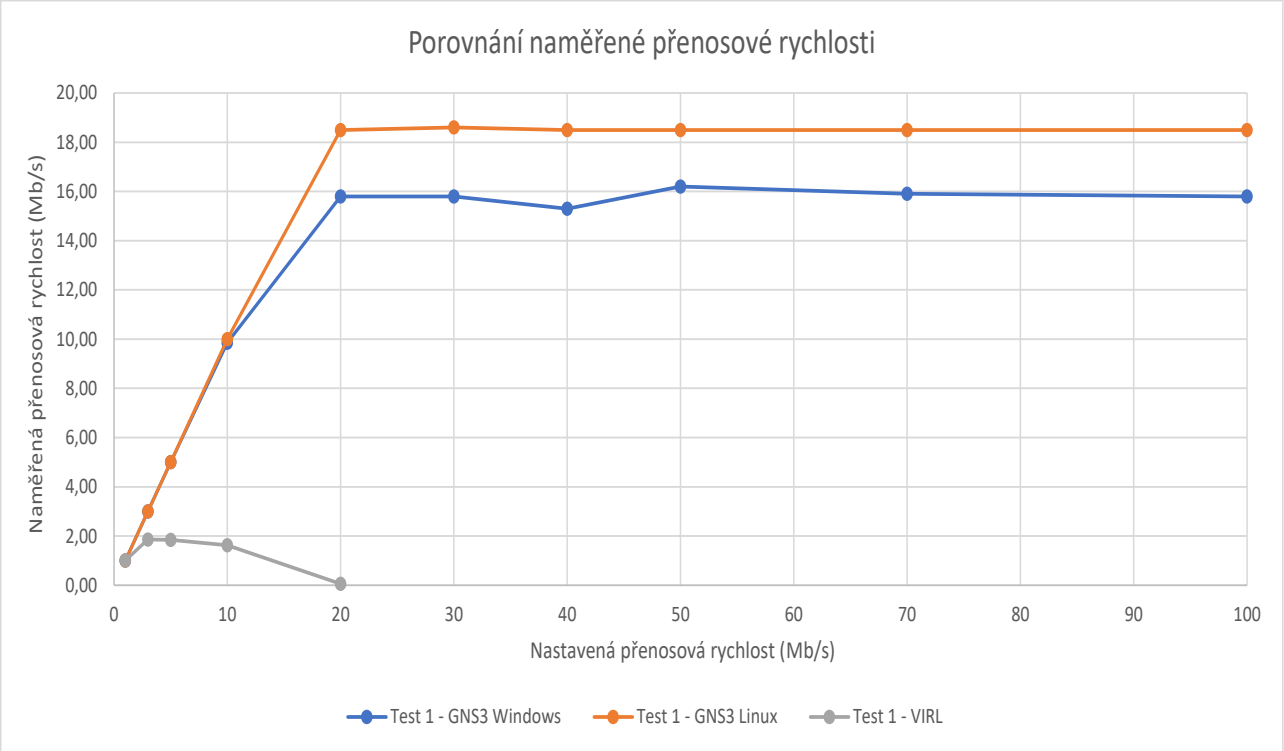
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	2,855	0/ 2552 (0%)
3	0,0-30,0	10,7	3,00	4,792	0/ 7650 (0%)
5	0,0-30,0	17,9	5,00	3,501	0/12744 (0%)
10	0,0-30,0	35,2	9,85	2,210	1/25129 (0%)
20	0,0-30,0	56,5	15,8	1,384	1/40272 (0%)
30	0,0-30,0	56,6	15,8	1,223	4/40389 (0,01%)
40	0,0-30,0	53,6	15,3	1,549	44/38301 (0,11%)
50	0,0-30,0	58,0	16,2	2,079	0/41342 (0%)
70	0,0-30,0	57,0	15,9	1,264	0/40648 (0%)
100	0,0-30,0	56,5	15,8	1,508	0/40323 (0%)

Tabulka 14: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Linux pro Test 2.1.

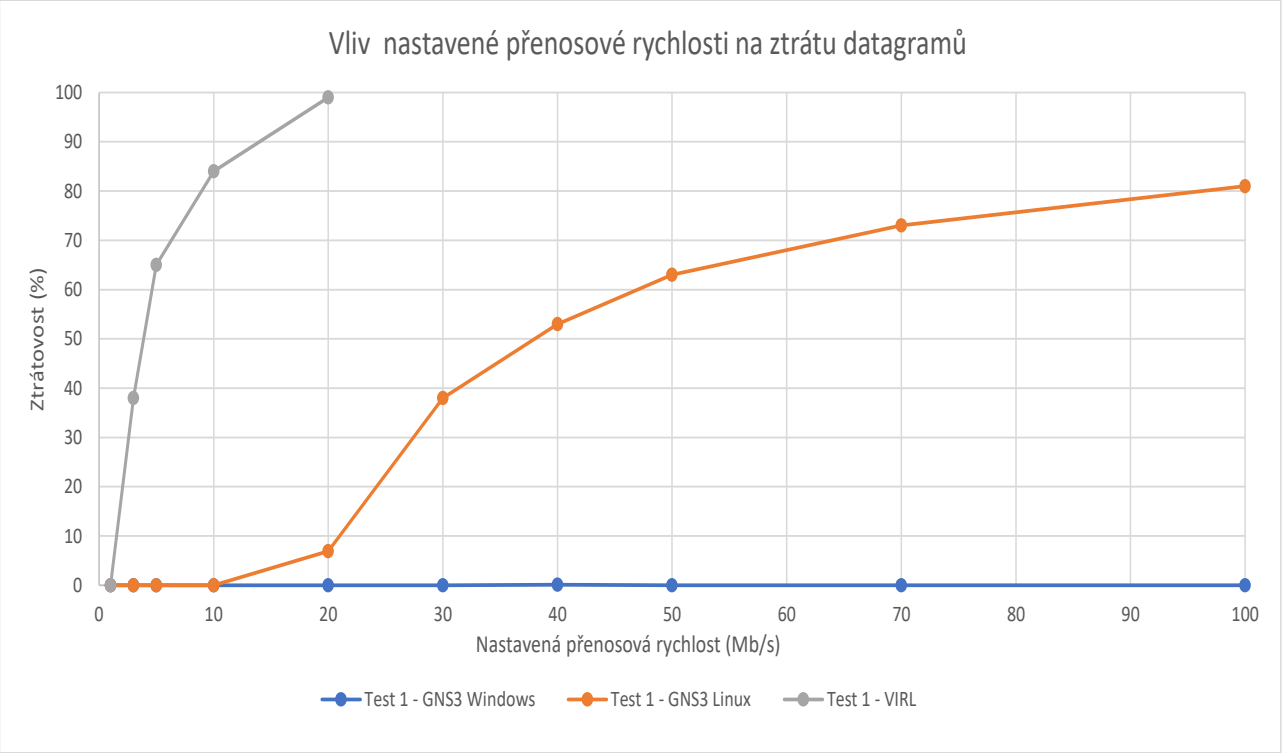
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	0,98	2,895	0/ 2552 (0%)
3	0,0-30,0	10,7	3,00	4,891	0/ 7653 (0%)
5	0,0-30,0	17,9	5,00	3,572	0/12756 (0%)
10	0,0-30,0	35,8	10,0	2,275	0/25511 (0%)
20	0,0-30,2	66,6	18,5	10,536	3504/51021 (6,9%)
30	0,0-30,1	66,6	18,6	0,949	29001/76531 (38%)
40	0,0-30,3	66,6	18,5	10,182	54528/102023 (53%)
50	0,0-30,3	66,6	18,5	10,131	80128/127635 (63%)
70	0,0-30,3	66,7	18,5	10,025	130963/178546 (73%)
100	0,0-30,3	66,6	18,5	10,372	208516/256020 (81%)

Tabulka 15: Naměřené hodnoty pro UDP datagramy v prostředí VIRL pro Test 2.1.

Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	1,278	0/ 2553 (0%)
3	0,0-30,3	6,70	1,86	16,473	2878/ 7654 (38%)
5	0,0-28,8	6,35	1,85	0,787	8227/12753 (65%)
10	0,0-30,2	5,82	1,62	16,761	21356/25511 (84%)
20	0,0-66,9	0,47	0,06	414.671	50689/51021 (99%)
30-100	Odpověď z iPerf serveru nedorazila.				



Obrázek 14: Porovnání naměřených přenosových rychlostí pro test 2.1 ve scénáři 2.



Obrázek 15: Zobrazené vlivu nastavené přenosové rychlosti na ztrátu datagramů pro test 2.1 ve scénáři 2.

Test TCP spojení

Tabulka 16: Test 2.1 pro TCP spojení ve scénáři 2.

Prostředí	Přeneseno (MB)	Přenosová rychlost Klient (Mb/s)	Přenosová rychlost Server (Mb/s)
GNS3 – Windows	23,2	19,2	18,3
GNS3 – Linux	22,6	19	18,1
VIRL	2,67	2,14	1,96

Testování odezvy pomocí nástroje PING

Tabulka 17: Testování odezvy pro test 2.1 ve scénáři 2.

Prostředí	Min(ms)	Průměr(ms)	Max(ms)	Odchylka(ms)	Ztráta(%)
GNS3 – Windows	12,70	17,20	23,40	2,87	0
GNS3 – Linux	11,191	16,234	21,443	3,031	0
VIRL	2,69	4,12	7,51	0,93	0

Vyhodnocení testu 2.1:

Jak je možné vidět v tabulce 15, pro hodnoty 30Mb/s až 100Mb/s nebylo možné v prostředí VIRL změřit přenosovou rychlost. Z tohoto důvudu není možné porovnat naměřené hodnoty s hodnotami naměřenými v prostředí GNS3. Z grafu na obrázku 14 můžeme říct, že nejvyšší přenosovou rychlost dosáhla simulační platforma GNS3 na OS Linux. Hodnota přenosové rychlosti konvergovala k hodnotě 18,5Mb/s. Při pozorování naměřených hodnot jitteru dopadlo simulační prostředí GNS3 na OS Linux, viz tabulka 14.

Z grafu na obrázku 15 je zřetelné, že k nejmenší ztrátě datagramů došlo v simulační platformě GNS3 na OS Windows. Přenos probíhal téměř bez ztrát datagramů.

Simulační platforma GNS3 v tomto testu dosáhla dobrých výsledku i v souvislosti s měřením TCP spojení na obou OS. V tabulce 16 si můžeme všimnout, že během testu bylo přeneseno přibližně 23MB a přenosová rychlost se na straně klienta pohybovala kolem 19Mb/s a na straně serveru 18Mb/s.

Při měření odezvy pomocí nástroje PING nejlépe dopadlo prostředí VIRL. Naměřené hodnoty v tabulce 17 ukazují, že průměrná doba odezvy dosáhla hodnoty 4,12ms.

Test 2.2

Test zaměřený na propustnost z oblasti 10 přes páteřní síť poskytovatele. Klientská stanice připojená do přepínače S6 umístěná v oblasti 10 generuje provoz na server, který je umístěn za směrovačem ABR4, tedy nahrazuje směrovač R6.

Testování přenosové rychlosti pomocí UDP

Tabulka 18: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Windows pro Test 2.2.

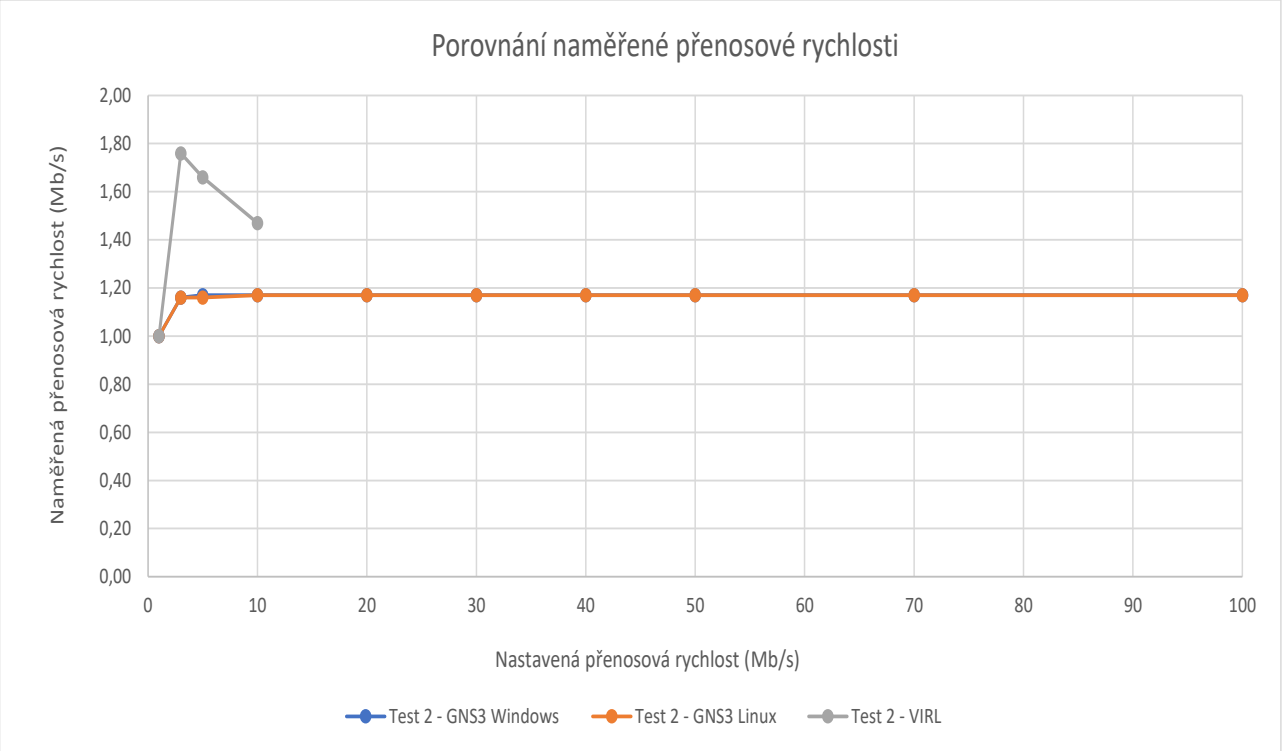
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	2,874	0/ 2552 (0%)
3	0,0-31,0	4,29	1,16	4,103	4595/ 7652 (60%)
5	0,0-31,0	4,31	1,17	3,907	9663/12740 (76%)
10	0,0-31,1	4,32	1,17	4,012	22249/25332 (88%)
20	0,0-31,1	4,34	1,17	3,207	37133/40231 (92%)
30	0,0-31,1	4,34	1,17	3,295	37577/40676 (92%)
40	0,0-31,2	4,36	1,17	4,124	36055/39165 (92%)
50	0,0-31,1	4,35	1,17	2,911	36819/39921 (92%)
70	0,0-31,1	4,35	1,17	2,269	37849/40949 (92%)
100	0,0-31,0	4,34	1,17	3,923	37183/40279 (92%)

Tabulka 19: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Linux pro Test 2.2.

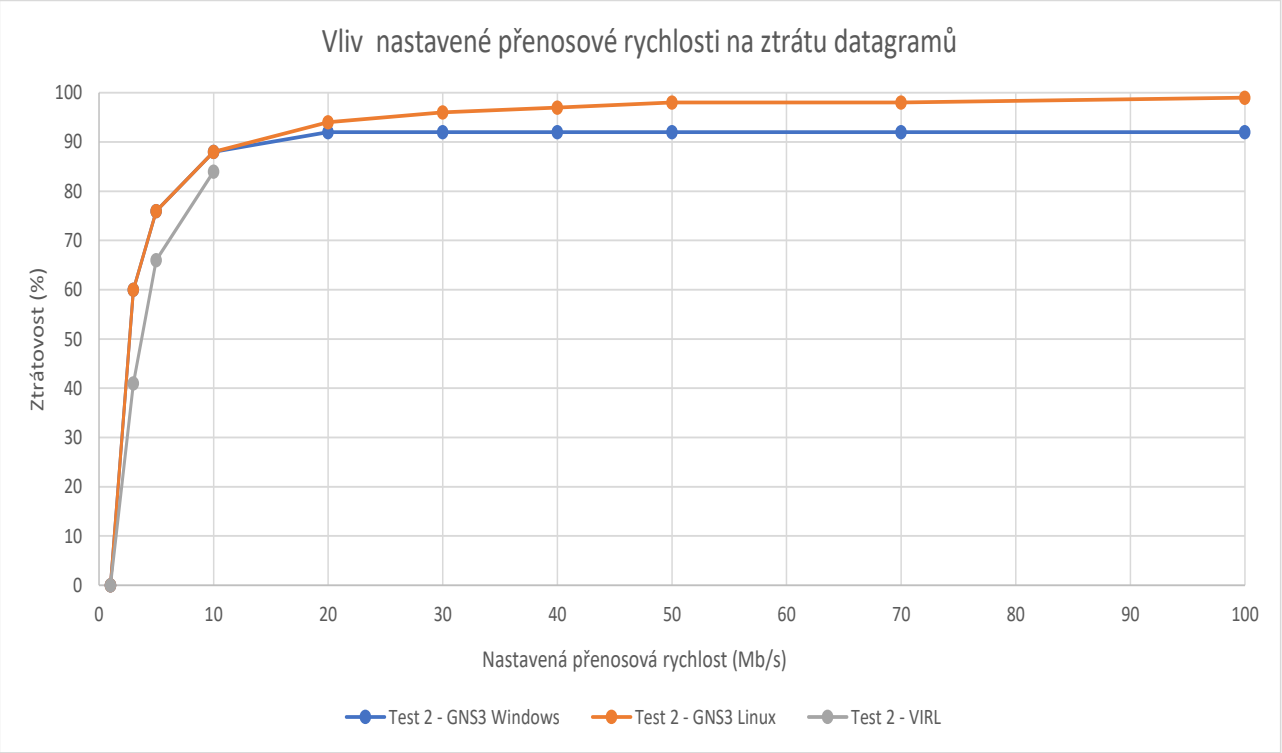
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	4,637	0/ 2552 (0%)
3	0,0-31,1	4,31	1,16	5,235	4579/ 7654 (60%)
5	0,0-31,0	4,31	1,16	4,573	9678/12752 (76%)
10	0,0-31,0	4,31	1,17	3,990	22433/25509 (88%)
20	0,0-31,0	4,31	1,17	3,575	47942/51018 (94%)
30	0,0-31,0	4,32	1,17	3,524	73446/76526 (96%)
40	0,0-31,0	4,32	1,17	3,025	98935/102013 (97%)
50	0,0-31,0	4,32	1,17	4,240	124562/127642 (98%)
70	0,0-31,1	4,32	1,17	4,110	175471/178550 (98%)
100	0,0-31,0	4,31	1,17	3,450	253057/256134 (99%)

Tabulka 20: Naměřené hodnoty pro UDP datagramy v prostředí VIRL pro Test 2.2.

Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	1,122	0/ 2553 (0%)
3	0,0-30,2	6,34	1,76	16,763	3131/ 7655 (41%)
5	0,0-30,2	5,99	1,66	17,030	8482/12755 (66%)
10	0,0-31,7	5,55	1,47	19,922	21552/25511 (84%)
20-100	Odpověď z iPerf serveru nedorazila.				



Obrázek 16: Porovnání naměřených přenosových rychlostí pro test 2.2 ve scénáři 2.



Obrázek 17: Zobrazené vlivu nastavené přenosové rychlosti na ztrátu datagramů pro test 2.2 ve scénáři 2.

Test TCP spojení

Tabulka 21: Test 2.2 pro TCP spojení ve scénáři 2.

Prostředí	Přeneseno (MB)	Přenosová rychlost Klient (Mb/s)	Přenosová rychlost Server (Mb/s)
GNS3 – Windows	2,12	1,65	1,14
GNS3 – Linux	2,12	1,66	1,14
VIRL	2,5	1,96	1,87

Testování odezvy pomocí nástroje PING

Tabulka 22: Testování odezvy pro test 2.2 ve scénáři 2.

Prostředí	Min(ms)	Průměr(ms)	Max(ms)	Odchylka(ms)	Ztráta(%)
GNS3 – Windows	73,69	87,66	94,85	5,22	0
GNS3 – Linux	52,544	63,552	80,596	7,192	0
VIRL	5,49	12,74	68,77	12,02	0

Vyhodnocení testu 2.2:

Z tabulky 20 je vidět, že testování přenosové rychlosti v prostředí VIRL nebylo možné pro hodnoty 20Mb/s až 100Mb/s. Pro tyto testované hodnoty nedorazila odpověď z iPerf serveru. Z tohoto důvodu není možné naměřené hodnoty v prostředí VIRL porovnávat s naměřenými hodnotami z prostředí GNS3. Dále z grafu na obrázku 16 můžeme říct, že naměřená přenosová rychlost a jitter v simulačním prostředí GNS3 byla totožná jak v OS Windows, tak Linux. Hodnota naměřené přenosové rychlosti konvergovala k hodnotě 1,17Mb/s, viz. graf na obrázku 17.

Při měření ztráty UDP datagramů nejmenší ztráty dosáhlo simulační prostředí GNS3 na OS Windows. Procentuální ztráta datagramů konverguje u vyšších přenosových rychlostí k 92%.

Testování TCP spojení a rychlosti odezvy vyznělo nejlépe pro prostředí VIRL. Jak je možné vidět v tabulce 21, při testování TCP spojení v prostředí VIRL bylo přeneseno 2,5MB dat a zároveň přenosová rychlost na straně klienta dosáhla hodnoty 1,96Mb/s a na straně serveru 1,87Mb/s. Pro měření doby odezvy v tabulce 22 průměrná doba odezvy dosáhla hodnoty 12,74ms.

Test 2.3

Test byl zaměřený na otestování propusnosti uvnitř oblasti 10, kde uživatelé jsou připojeni do různých VLAN a chtějí mezi sebou komunikovat. Jelikož L2 vrstva nemá žádné mechanismy, které by umožňovali vyměnit si zprávy mezi sebou z různých VLAN, bylo nutné použít zařízení z L3 vrstvy a nakonfigurovat tzv. Router on the stick. Klientská stanice byla připojena do přepínače S6 a server do přepínače S7.

Testování přenosové rychlosti pomocí UDP

Tabulka 23: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Windows pro Test 2.3.

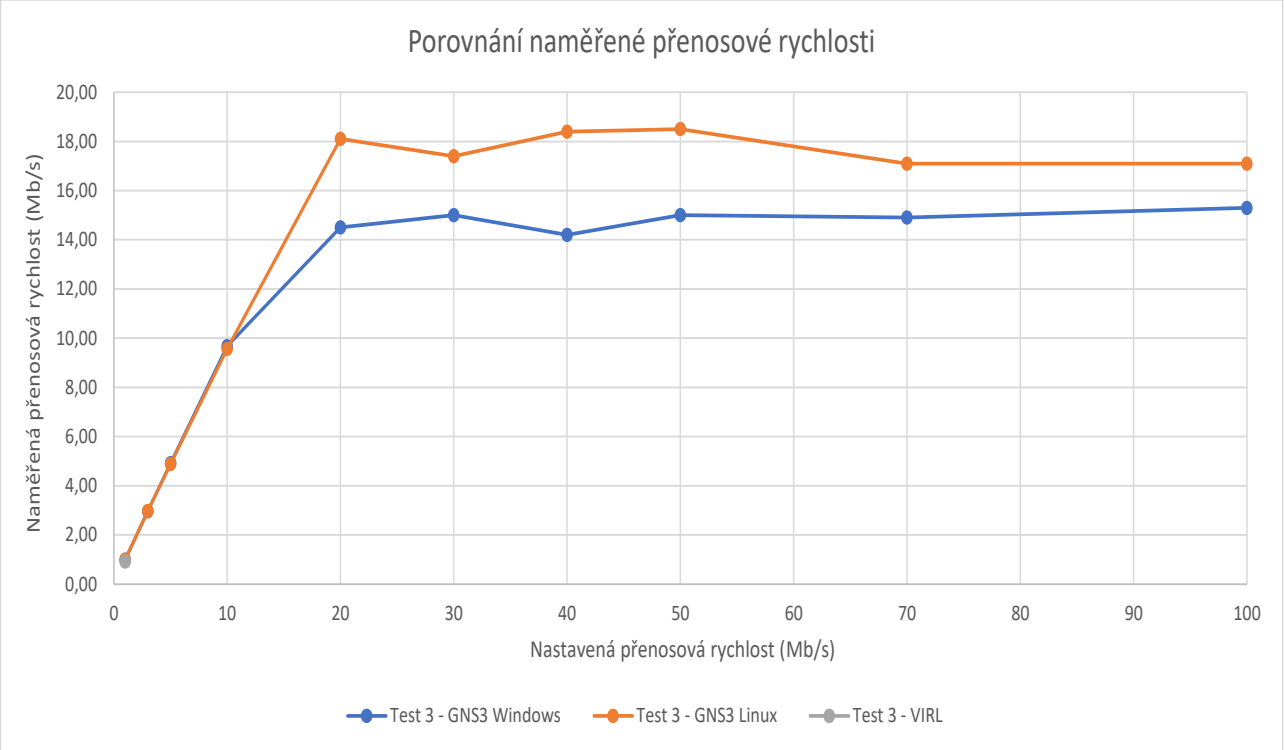
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,57	0,98	2,574	2/ 2552 (0,078%)
3	0,0-30,0	10,6	2,97	4,802	74/ 7653 (0,97%)
5	0,0-30,0	17,6	4,92	3,518	189/12748 (1,5%)
10	0,0-30,0	34,6	9,68	1,917	608/25297 (2,4%)
20	0,0-30,0	52,0	14,5	1,445	1040/38132 (2,7%)
30	0,0-30,0	53,5	15,0	1,147	1028/39219 (2,6%)
40	0,0-30,0	50,8	14,2	1,392	349/36583 (0,95%)
50	0,0-30,0	53,5	15,0	1,399	1031/39199 (2,6%)
70	0,0-30,0	53,1	14,9	1,428	1099/38985 (2,8%)
100	0,0-30,0	54,7	15,3	1,960	753/39783 (1,9%)

Tabulka 24: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Linux pro Test 2.3.

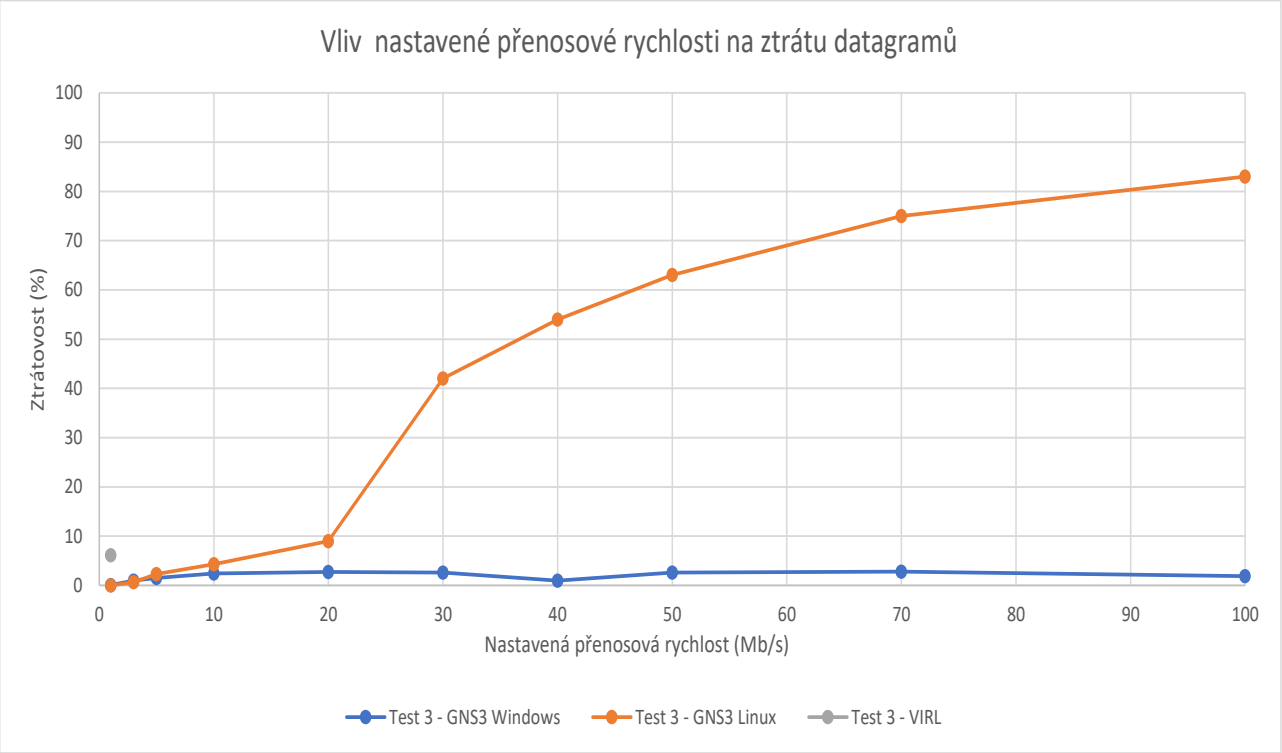
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	0,98	2,990	0/ 2552 (0%)
3	0,0-30,0	10,7	2,98	4,716	53/ 7653 (0,69%)
5	0,0-30,0	17,5	4,88	3,609	299/12756 (2,3%)
10	0,0-30,0	34,2	9,57	2,004	1096/25512 (4,3%)
20	0,0-30,1	65,1	18,1	1,118	4594/51020 (9%)
30	0,0-30,1	62,4	17,4	1,431	32009/76531 (42%)
40	0,0-30,3	66,2	18,4	10,348	54828/102035 (54%)
50	0,0-30,1	66,5	18,5	1,429	80226/127635 (63%)
70	0,0-30,2	61,5	17,1	10,230	134654/178524 (75%)
100	0,0-30,1	61,3	17,1	1,502	212325/256083 (83%)

Tabulka 25: Naměřené hodnoty pro UDP datagramy v prostředí VIRL pro Test 2.3.

Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,36	0,92	1,166	155/ 2553 (6,1%)
3-100	Odpověď z iPerf serveru nedorazila.				



Obrázek 18: Porovnání naměřených přenosových rychlostí pro test 2.3 ve scénáři 2.



Obrázek 19: Zobrazené vlivu nastavené přenosové rychlosti na ztrátu datagramů pro test 2.3 ve scénáři 2.

Test TCP spojení

Tabulka 26: Test 2.3 pro TCP spojení ve scénáři 2.

Prostředí	Přeneseno (MB)	Přenosová rychlost Klient (Mb/s)	Přenosová rychlost Server (Mb/s)
GNS3 – Windows	19,4	15,8	14,9
GNS3 – Linux	20,04	16,8	15,9
VIRL	1,3	1,09	0,87

Testování odezvy pomocí nástroje PING

Tabulka 27: Testování odezvy pro test 2.3 ve scénáři 2.

Prostředí	Min(ms)	Průměr(ms)	Max(ms)	Odchylka(ms)	Ztráta(%)
GNS3 – Windows	11,73	18,22	32,40	4,09	0
GNS3 – Linux	11,02	16,002	20,881	2,934	0
VIRL	4,71	8,25	45,47	5,65	0

Vyhodnocení testu 2.3:

Při testování přenosové rychlosti v prostředí VIRL nastal totožný problém jako v předchozím scénáři. Nebylo možné provést test pro všechny požadované hodnoty nastavené přenosové rychlosti. Proto jsem získanou hodnotu neporovnával s výsledky z GNS3. Jak je možné vidět z tabulky 25 podařilo se otestovat přenosovou rychlost pouze pro 1Mb/s. Z grafu na obrázku 16 vidíme, že nejvyšší naměřená přenosová rychlost byla v prostředí GNS3 na platformě Linux. Hodnota naměřené přenosové rychlosti konvergovala k hodnotě 17Mb/s. Z tabulky 24 je vidět, že jitter v simulačním prostředí GNS3 dosahlo horších hodnot na OS Linux.

Při měření ztráty datagramů nejlépe dopadlo simulační prostředí na platformě Windows, jak je možné vidět z grafu na obrázku 19, naměřená ztráta konverguje u vyšších přenosových rychlostí k 2%.

Testování TCP spojení dopadlo nejlépe v simulačním prostředí GNS3 na OS Linux. Jak je možné vidět v tabulce 26, při testování TCP spojení bylo přeneseno 20MB a naměřená přenosová rychlost na straně klienta dosáhla 16,8Mb/s a na straně serveru 15,9Mb/s.

Testování rychlosti odezvy nejlépe dopadlo v profesionálním řešením VIRL. Z tabulky 27 je vidět, že průměrná doba odezvy v tomto prostředí dosáhla hodnoty 8,25ms.

7.3 Scénář 3

Test 3.1

Test byl zaměřený na otestování propusnosti technologie MPLS – VPN, ze zákaznické sítě S16 přes páteřní síť poskytovatele do sítě zákazníka S13. Klientská stanice byla připojena do zákaznického směrovače CE3 v zákaznické síti S13 a generovala provoz na server, který byl umístěn v síti S16 a připojen do zákaznického směrovače CE1. Jednalo se tedy o testování propustnosti celkového navrhnutého řešení.

Testování přenosové rychlosti pomocí UDP

Tabulka 28: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Windows pro Test 3.1.

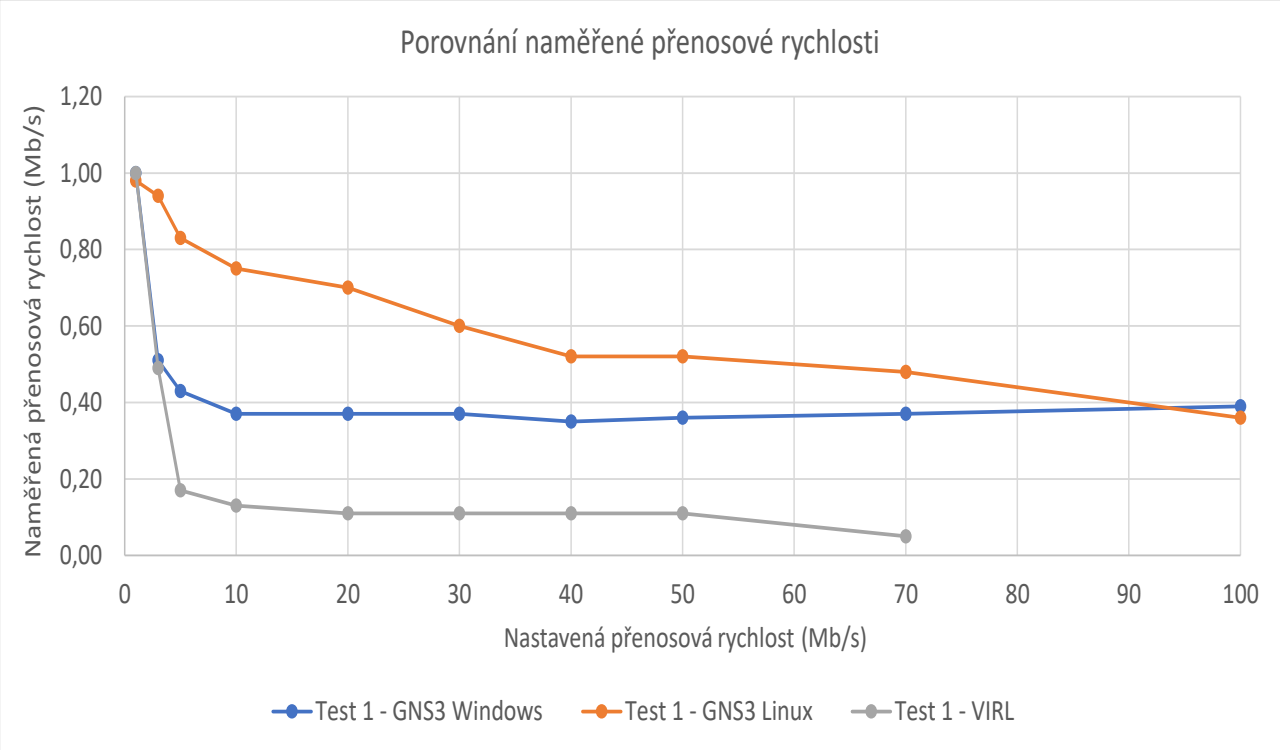
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	3,156	0/ 2551 (0%)
3	0,0-30,8	1,93	0,51	3,813	6230/ 7608 (82%)
5	0,0-30,8	1,62	0,43	3,495	11501/12654 (91%)
10	0,0-31,0	1,39	0,37	5,769	13750/14739 (93%)
20	0,0-30,8	1,38	0,37	5,404	14911/15893 (94%)
30	0,0-30,7	1,38	0,37	3,778	14646/15630 (94%)
40	0,0-30,8	1,32	0,35	5,516	14002/14942 (94%)
50	0,0-30,8	1,36	0,36	4,088	15087/16055 (94%)
70	0,0-30,7	1,38	0,37	3,728	15289/16275 (94%)
100	0,0-30,8	1,47	0,39	3,825	13321/14370 (93%)

Tabulka 29: Naměřené hodnoty pro UDP datagramy v prostředí GNS3 na platformě Linux pro Test 3.1.

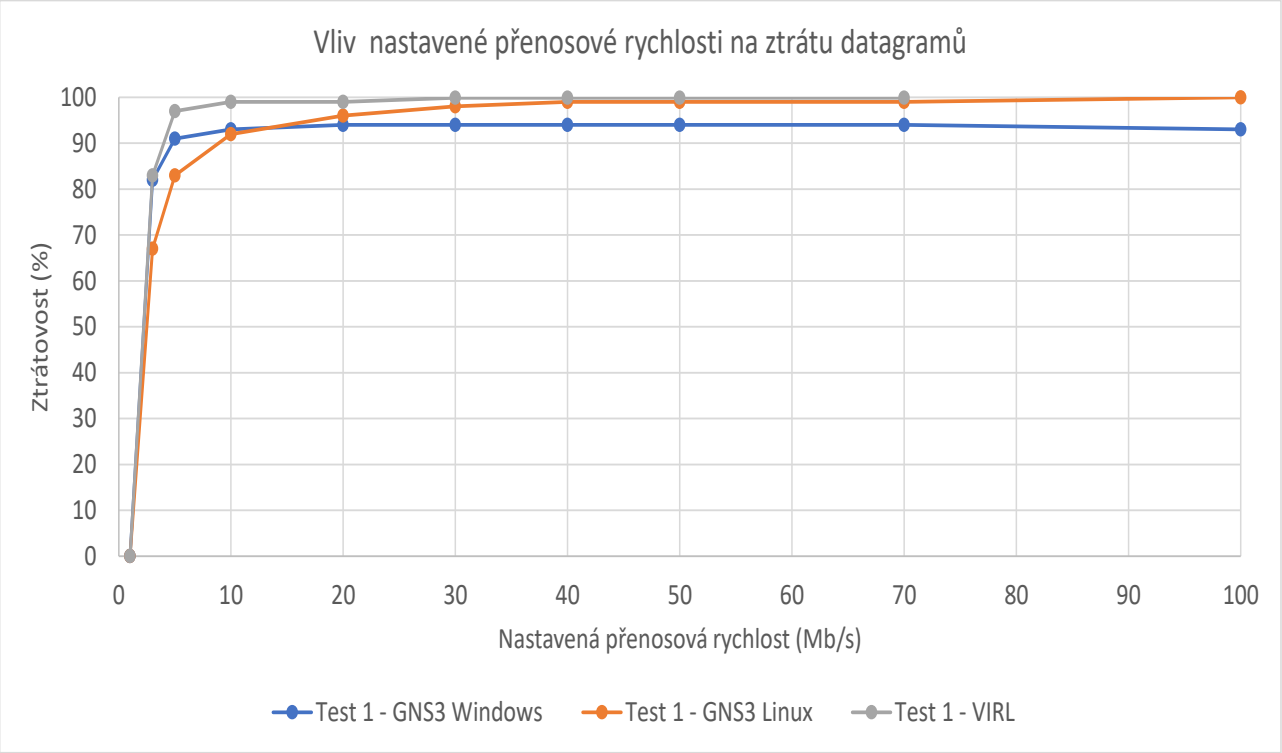
Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	0,98	2,751	0/ 2552 (0%)
3	0,0-30,7	3,53	0,94	3,138	5134/ 7653 (67%)
5	0,0-30,8	3,10	0,83	1,491	10543/12755 (83%)
10	0,0-30,8	2,80	0,75	2,801	23512/25511 (92%)
20	0,0-30,7	2,62	0,70	2,633	49145/51015 (96%)
30	0,0-30,8	2,26	0,60	3,290	74887/76497 (98%)
40	0,0-30,8	1,96	0,52	3,312	100579/101980 (99%)
50	0,0-30,7	1,96	0,52	2,696	126137/127532 (99%)
70	0,0-30,8	1,82	0,48	2,415	175344/176639 (99%)
100	0,0-30,9	1,37	0,36	5,908	226295/227270 ($\approx$ 100%)

Tabulka 30: Naměřené hodnoty pro UDP datagramy v prostředí VIRL pro Test 3.1.

Nastavená přenosová rychlost (Mb/s)	Interval (s)	Přeneseno (MB)	Naměřená přenosová rychlost (Mb/s)	Jitter (ms)	Ztraceno/Celkem datagramů
1	0,0-30,0	3,58	1,00	0,676	0/ 2553 (0%)
3	0,0-31,0	1,85	0,49	64,037	6333/ 7654 (83%)
5	0,0-31,0	0,63	0,17	63,595	12308/12754 (97%)
10	0,0-30,5	0,48	0,13	32,504	25158/25501 (99%)
20	0,0-31,0	0,43	0,11	63,582	50703/51007 (99%)
30	0,0-30,5	0,41	0,11	32,130	76030/76325 (≈ 100%)
40	0,0-30,7	0,40	0,11	47,492	101735/102023 (≈ 100%)
50	0,0-30,7	0,41	0,11	47,382	127315/127605 (≈ 100%)
70	0,0-69,2	0,40	0,05	117,669	177782/178067 (≈ 100%)
100	Odpověď z iPerf serveru nedorazila.				



Obrázek 20: Porovnání naměřených přenosových rychlostí pro test 3.1 ve scénáři 3.



Obrázek 21: Zobrazené vlivu nastavené přenosové rychlosti na ztrátu datagramů pro test 3.1 ve scénáři 3.

Test TCP spojení

Tabulka 31: Test 3.1 pro TCP spojení ve scénáři 3.

Prostředí	Přeneseno (MB)	Přenosová rychlost Klient (Mb/s)	Přenosová rychlost Server (Mb/s)
GNS3 – Windows	2,12	1,61	1,14
GNS3 – Linux	2,12	1,59	1,14
VIRL	2,74	2,25	2,15

Testování odezvy pomocí nástroje PING

Tabulka 32: Testování odezvy pro test 3.1 ve scénáři 3.

Prostředí	Min(ms)	Průměr(ms)	Max(ms)	Odchylka(ms)	Ztráta(%)
GNS3 – Windows	51,15	72,15	100,83	12,51	0
GNS3 – Linux	51,13	65,83	96,19	10,67	0
VIRL	6,75	20,35	72,31	12,7	0

Vyhodnocení testu 3.1:

Z grafu na obrázku 20 můžeme vidět, že s nastavenou přenosovou rychlostí klesá naměřená přenosová rychlost. Dále můžeme říct, že nejpomaleji klesá přenosová rychlost v prostředí GNS3 na OS Linux. U profesionálního řešení VIRL si můžeme všimnout z tabulky 30, že pro nastavenou přenosovou rychlost 100Mb/s se nepodařilo zjistit naměřenou přenosovou rychlost a to z důvodu, že iPerf server nedokázal iPerf klientovi poslat odpoveď s výsledky. Dále si můžeme všimnout, že v prostředí VIRL byl naměřen daleko větší jitter, než je tomu v prostředí GNS3.

Dále z grafu na obrázku 21 je možné vidět, že nejmenší ztrátu datagramů má platforma GNS3 na OS Windows, kde hodnota ztráty datagramů konverguje u vyšších přenosových rychlostí k 93%.

Při měření TCP spojení nejlepších výsledků dostahuje prostředí VIRL. Dle tabulky 31 se v prostředí VIRL přeneslo 2,74MB a byla naměřena přenosová rychlost na straně klienta 2,25Mb/s a na straně serveru 2,15Mb/s.

Testování odezvy dopadlo opět ve prospěch prostředí VIRL. Jak je vidět v tabulce 32, průměrná doba odezvy v prostředí VIRL je rovna hodnotě 20,35%.

Test 3.2

Test měl ověřit propustnost technologie AToM. Klientská stanice byla připojena do přepínače SW2 a měla generovat provoz na server připojený do přepínače SW1. Testy pomoci nástroje iPerf nebylo možné uskutečnit. Nástroj nedokázal navázat spojení a začít generovat provoz. Tento problém se vyskytl jak v simulačním prostředí GNS3, tak na profesionálním řešením VIRL. Podařilo se tedy otestovat pouze rychlost odezvy pomocí nástroje PING.

Testování odezvy pomocí nástroje PING

Tabulka 33: Testování odezvy pro test 3.2 ve scénáři 3.

Prostředí	Min(ms)	Průměr(ms)	Max(ms)	Odchylka(ms)	Ztráta(%)
GNS3 – Windows	37,78	48,27	60,85	9,3	0
GNS3 – Linux	30,867	36,479	50,21	4,709	0
VIRL	4,02	5,74	14,09	1,54	0

Vyhodnocení testu 3.2:

Jak je možné vidět z tabulky 33, nejlepších výsledků dosáhlo prostředí VIRL. Průměrná doba odezvy je v tomto případě rovna 5,74ms.

7.4 Paměťové a výpočetní nároky

Test 1

Cílem testu bylo ověřit využití paměti RAM a procesoru jednotlivými procesy dynamipsu, GNS3 a GNS3 serveru, tedy kompletního řešení po spuštění jednotlivých scénářů. Nutno podotknout, že test probíhal bez připojení koncových stanic, pouze se směrovači a přepínači. K tomuto účelu byl vytvořen testovací skript ve skriptovacím jazyku Powershell [11]. Powershell je možné použít jak na platformě Windows, tak nově i na Linuxu. Testovací skript byl spuštěn zároveň se scénářem a jeho funkce je následující:

```
for($i=1; $i -le 60; $i++){

    Get-Process -pid 8671 | Select-Object @{Name='RAM(MB)';Expression={($_.
        WorkingSet64/1024kb)}}, @{Name='CPU(s)';Expression={($_.cpu)}} | Export-
        Csv -Path "C:\Users\Marek\Desktop\PID.csv" -Delimiter ";" -Append
                                .
                                .
                                .
    Get-Process -pid PID | Select-Object @{Name='RAM(MB)';Expression={($_.
        WorkingSet64/1024kb)}}, @{Name='CPU(s)';Expression={($_.cpu)}} | Export-
        Csv -Path "C:\Users\Marek\Desktop\gns3server.csv" -Delimiter ";" -Append

    Start-Sleep -s 1
}
```

Výpis 37: Ukázka testovacího skriptu pro test 1.

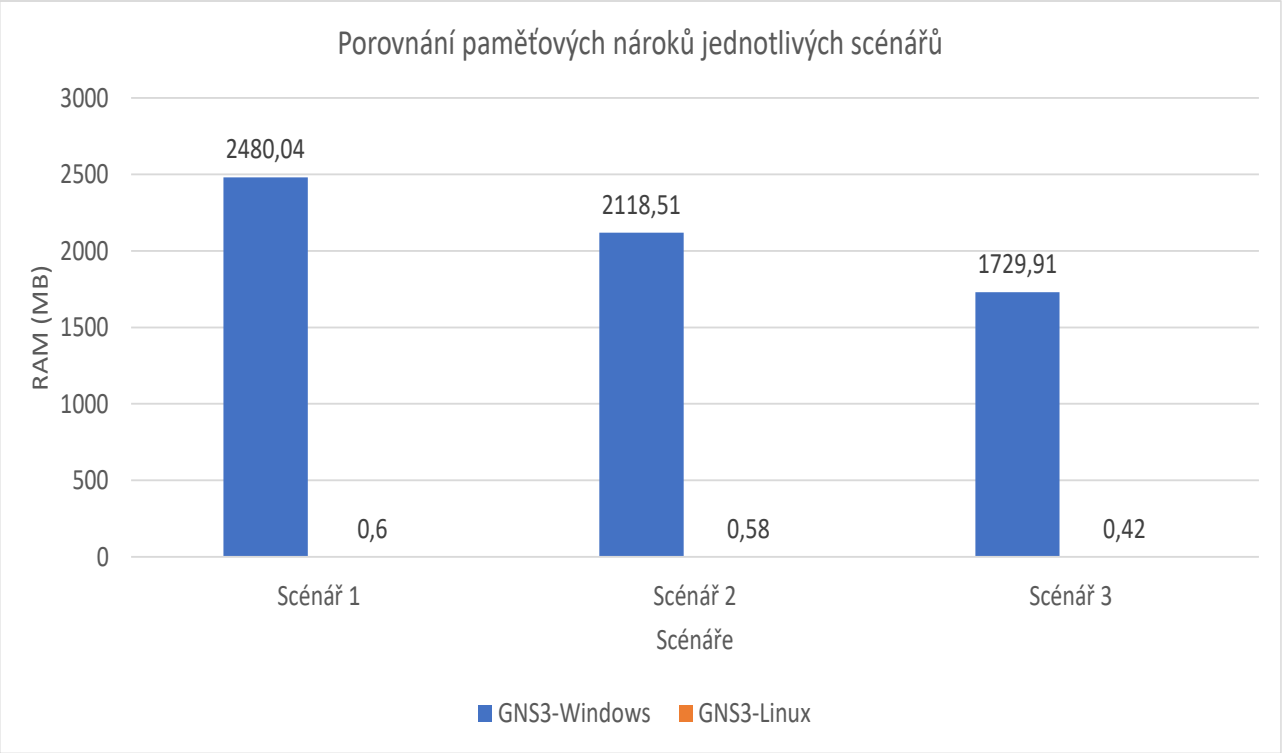
Po dobu jedné minuty od spuštění scénáře, skript každou sekundu zaznamenává pro jednotlivé PID hodnoty WorkingSet64 (MB) a CPU (s). Naměřené hodnoty ukládá do souborů, které jsou pojmenované podle jednotlivých PID. Pro každý PID byl vytvořen průměr z naměřených hodnot (60 hodnot pro každé PID) a tyto hodnoty jsou čtenáři k dispozici v následujících tabulkách a také vizualizovány ve formě skupinových sloupcových grafů.

Tento test bylo možné aplikovat pouze v simulační platformě GNS3 (Windows i Linux) a to z důvodu, že prostředí VIRL neposkytuje mechanismy jak měřit RAM a CPU pro jednotlivé instance PID, ale poskytuje pouze grafické znázornění využití RAM celkového řešení a počet použitých vCPU.

Kompletní naměřené hodnoty jsou součástí příloh v adresáři Paměťové a výpočetní nároky-/Test1 a jedná se o soubory linux.xlsx a windows.xlsx.

Tabulka 34: Tabulka součtu průměrů jednotlivých měření paměti RAM

	Scénář1	Scénář2	Scénář3
Prostředí	RAM(MB)	RAM(MB)	RAM(MB)
GNS3 – Windows	2480,04	2118,51	1729,91
GNS3 – Linux	0,6	0,58	0,42

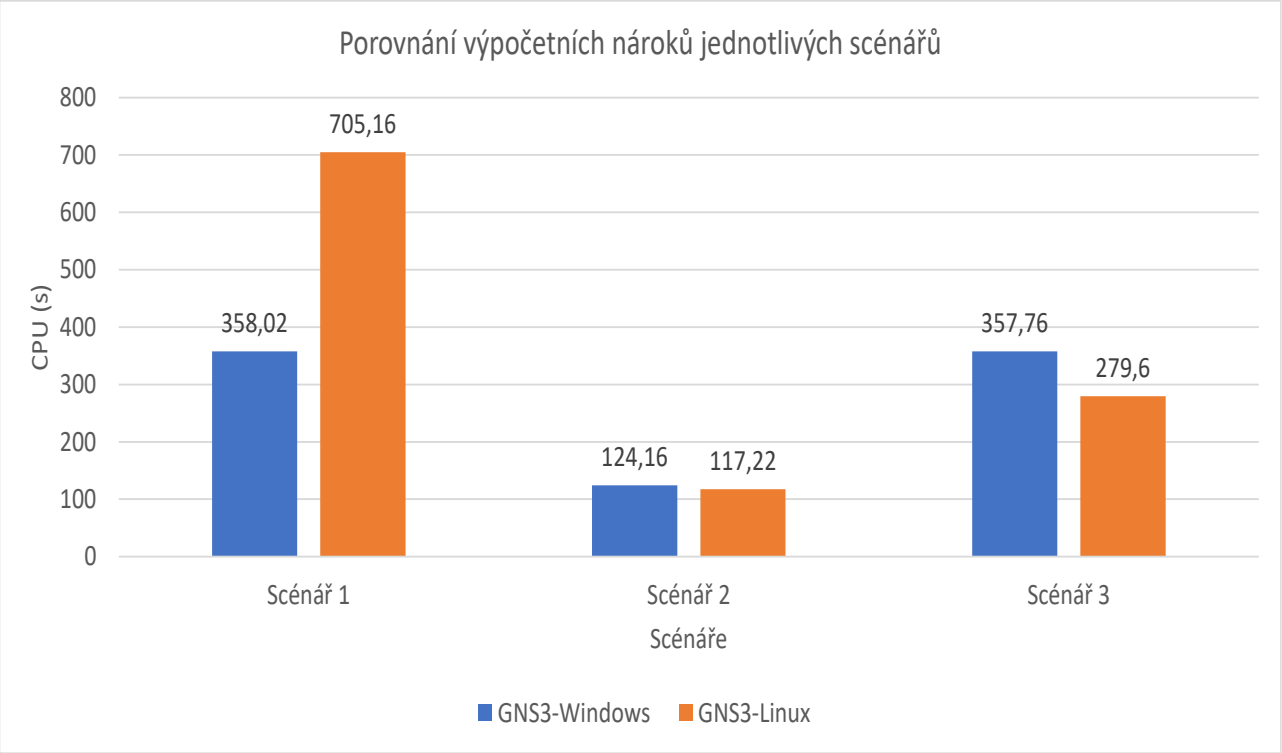


Obrázek 22: Paměťové nároky pro jednotlivé scénáře spuštěné v prostředí GNS3.

Z grafu na obrázku 22 můžeme říct, že spotřeba RAM paměti jednotlivých scénářů v simulačním prostředí GNS3 pro test 1 dopadla nejhůře na OS Windows. Naměřené hodnoty pomocí testovacího skriptu v OS Linux jsou velmi nízké. Tento fakt může být způsoben chováním dynamipsu v prostředí Linux. Jednolivé procesy dynamips zřejmě nepotřebují pro svou práci velké množství paměti RAM na OS Linux.

Tabulka 35: Tabulka součtu průměrů jednotlivých měření CPU

	Scénář1	Scénář2	Scénář3
Prostředí	CPU(s)	CPU(s)	CPU(s)
GNS3 – Windows	358,02	124,16	357,76
GNS3 – Linux	705,16	117,22	279,6



Obrázek 23: Výpočetní nároky pro jednotlivé scénáře spuštěné v prostředí GNS3.

Z grafu na obrázku 23 můžeme vidět, že čas potřebný k provedení simulace scénáře 1 v prostředí Linux jsou vyšší o 347s, než na OS Windows. Scénář 2 dopadl o přibližně 7s lépe ve prospěch OS Linux a scénář 3 dopadl o 78s hůře na OS Windows.

Test 2

Jelikož Test1 byl zaměřen na měření paměti RAM a CPU po dobu první minuty od spuštění scénáře, rozhodl jsem se, že Test2 provedu pro měření až po konvergenci všech použitých protokolů.

Pro účely testu byl opět vytvořen testovací skript v powershellu pro prostředí GNS3 a hodnota paměti RAM byla v prostředí VIRL odečtena z grafického znázornění využití RAM. V obou případech došlo k odečtení hodnot po 180 sekundách, což považuji za dostatečně bezpečný čas pro konvergenci všech nakonfigurovaných protokolů.

```
sleep(180)
Get-Process -Name dynamips | Select-Object @{Name='PID';Expression={$_.ID}}, @{
    Name='RAM(MB)';Expression={($_.WorkingSet64/1024kb)}}, @{Name='CPU(s)';
    Expression={($_.cpu)}}
```

Výpis 38: Ukázka testovacího skriptu pro test 2.

Měření v GNS3 probíhalo pouze na jednotlivých instancích dynamipsu a bez koncových stanic. V prostředí VIRL tomu bylo podobně, do topologií jsem umístil pouze směrovače a přepánače.

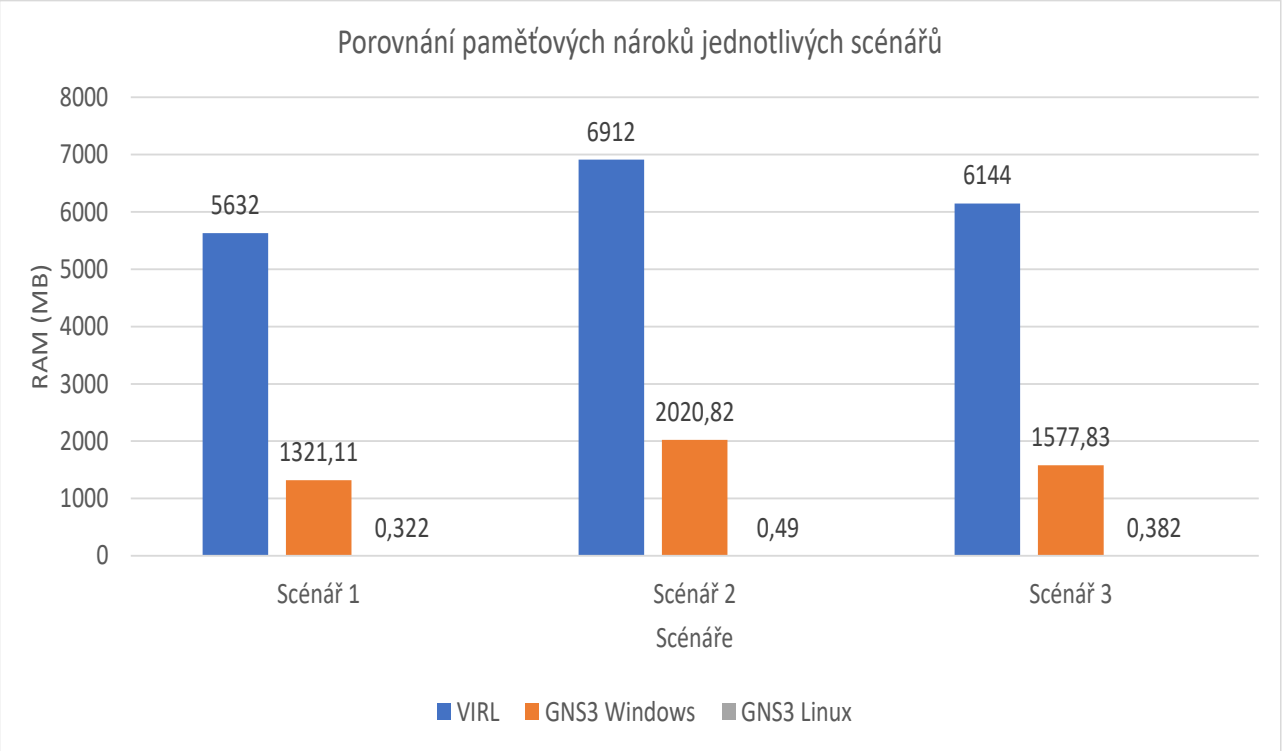
Prostředí VIRL neumožňuje spustit celý scénář 1 tak jak byl navržen, z důvodu nízké kapacity vCPU. Umožňuje tedy spustit scénáře s maximálně 11 instancemi přepínačů, nebo směrovačů. Tento fakt byl zohledněn a přizpůsoben v GNS3 tak, abych opravdu testoval totožné topologie jako v prostředí VIRL. Scénář 1 byl tedy omezen na dva přepínače (SW1, SW3) a směrovače (R7, CE5, CE6, PE1, PE2, CE9, CE10, R11). Ve scénáři 2 byl totožný problém, nedostatek vCPU, ale zde stačilo ubrat jeden přepínač SW5. Scénář 3 zůstal beze změn.

Další omezení v prostředí VIRL spočívalo v tom, že VIRL umožňuje pouze grafické znázornění využití RAM a počet použitých vCPU. Z tohoto důvodu je tento test provonatelný s GNS3 pouze pro paměť RAM, ale pro zajímavost uvádím i počet vCPU nutných pro simulaci jednotlivých scénářů v prostředí VIRL.

Kompletní naměřené hodnoty jsou součástí příloh v adresáři Paměťové a výpočetní nároky-/Test2 a jedná se o soubor test2.xlsx.

Tabulka 36: Tabulka využití paměti RAM jednotlivých scénářů v prostředí GNS3 a VIRL po konvergenci sítě.

	Scénář1	Scénář2	Scénář3
Prostředí	RAM(MB)	RAM(MB)	RAM(MB)
VIRL	5632	6912	6144
GNS3 – Windows	1321,11	2020,82	1577,83
GNS3 – Linux	0,322	0,49	0,38

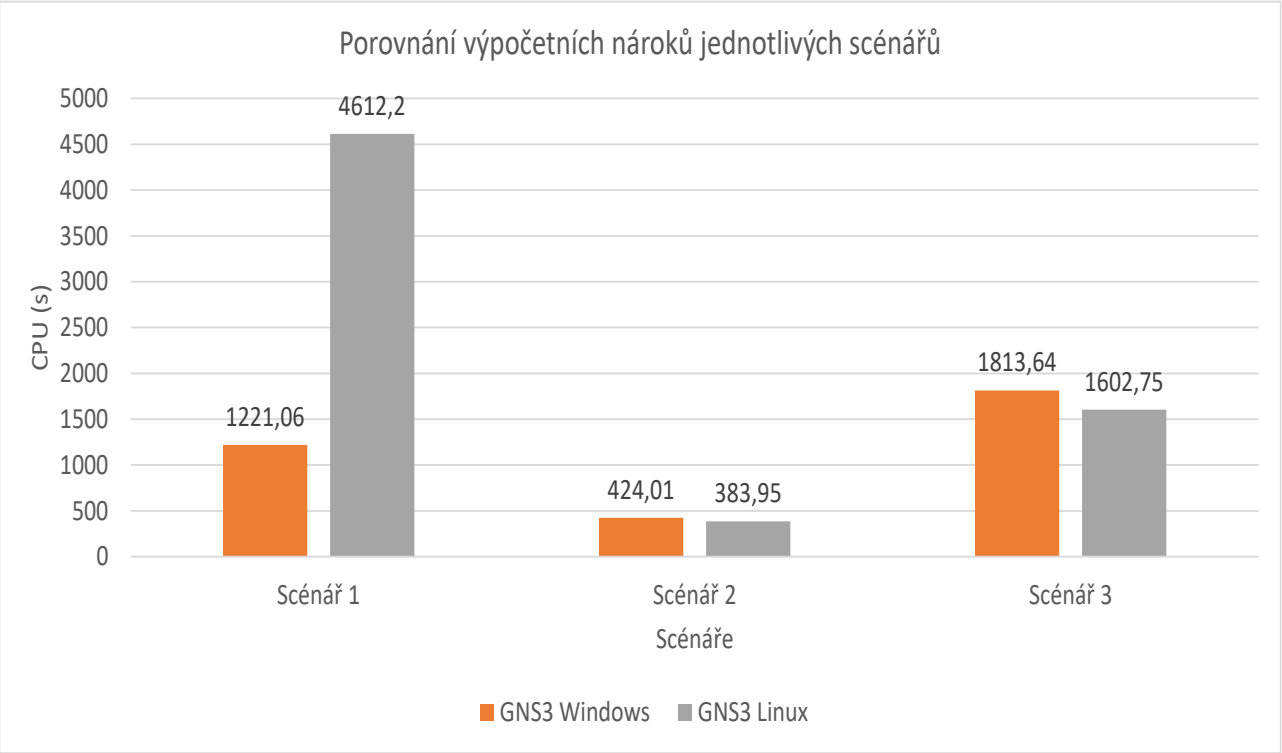


Obrázek 24: Paměťové nároky pro jednotlivé scénáře spuštěné v prostředí GNS3 a VIRL.

Z grafu na obrázku 24 je zřejmé, že největší paměťové nároky má profesionální řešení VIRL. Nejlépe dopadlo simulační prostředí GNS3 na platformě Linux.

Tabulka 37: Tabulka využití CPU jednotlivými scénáři v prostředí GNS3 a VIRL po konvergenci sítě.

	Scénář1	Scénář2	Scénář3
Prostředí	CPU(s)	CPU(s)	CPU(s)
GNS3 – Windows	1221,06	424,01	1813,64
GNS3 – Linux	4612,2	383,95	1602,75



Obrázek 25: Výpočetní nároky pro jednotlivé scénáře spuštěné v prostředí GNS3.

Srovnání výpočetních nároků v prostředí GNS3 pro test 2, který byl zaměřen na měření po konvergenci sítě dopadl podobně jako test 1, který měřil výpočetní nároky během první minuty od spuštění jednotlivých scénářů. Jak je možné vidět na grafu z obrázku 25, scénář 1 dopadl nejhůře na OS Linux o 3391s oproti nasazení na OS Windows. Hodnoty naměřené v scénáři 2 pro OS Linux jsou lepší o 40s oproti OS Windows. Scénář 3 dopadl lépe o 211s na OS Linux.

Tabulka 38: Tabulka počtu vCPU nutných pro spuštění jednotlivých scénářů v prostředí VIRL.

	Scénář1	Scénář2	Scénář3
Prostředí	vCPU	vCPU	vCPU
VIRL	10	11	11

Tabulka 38 uvádí kolik vCPU vyžadovalo profesionální řešení pro jednotlivé scénáře. Školní licence sice umožňuje mít spuštěno až 20 instancí zařízení současně, ale stále je práce v prostředí VIRL omezená hardwarem serveru, na kterém je VIRL spušťen.

8 Závěr

Tato diplomová práce je zaměřená na testování dostupných simulačních prostředí. Hlavním cílem diplomové práce bylo porovnat existující nekomerční simulační prostředí GNS3 s profesionálním řešením VIRL od firmy Cisco, které je založeno na platformě OpenStack.

V úvodu teoretické části mé práce jsem se zaměřil na představení simulačních prostředí GNS3 a VIRL. Dále jsem popsal OpenStack, jakožto platformu, na které je založené profesionální řešení VIRL.

U testování simulační platformy GNS3 jsem využil multiplatformnost tohoto nástroje a veškeré otestování jsem provedl jak na operačním systému Microsoft Windows 10 Home 64bit (verze 1607, build operačního systému 14393.1066), tak na operačním systému Linux Ubuntu 16.04 LTS Xenial Xerus 64bit (verze kernelu: 4.4.0-72-generic). K profesionálnímu řešení VIRL jsem měl přístup v rámci univerzity a testování probíhalo v aplikaci VM Maestro, která je dostupná pro práci s prostředím VIRL.

Pro účely srování konfigurovatelnosti obou simulačních prostředí jsem navrh a nakonfiguroval tři scénáře typických simulovaných sítí. Scénáře jsou nakonfigurované na IPv4, nicméně se domnívám, že na IPv6 bych dosáhl podobných výsledků. Dále jsem nad těmito scénáři prováděl testy, které byly zaměřeny na propustnost, výkon jednotlivých prvků a výkon celkového řešení, dále paměťové a výpočetní nároky. K otestování propustnosti a výkonu jednotlivých síťových prvků jsem použil nástroj iPerf a PING.

Další kategorie testů byla zaměřena na možnosti zachycení provozu v jednotlivých simulačních prostředí. K tomuto účelu byl v prostředí GNS3 použit paketový analyzátor Wireshark a nástroj tcpdump. V profesionálním řešení byl zachytáván provoz přímo prostředím VIRL do souborového formátu pcap, se kterým je dále možné pracovat v programu Wireshark, nebo tcpdump.

K testování paměťových a výpočetních nároků kladených na provoz obou řešení jsem použil testovací skript napsaný ve skriptovacím jazyce PowerShell. Naměřené hodnoty jednotlivých testů jsem v této práci prezentoval ve formě tabulek a grafů.

Ze získaných poznatků mé práce mohu říct, že simulační platforma GNS3 má menší paměťové a výpočetní nároky než profesionální řešení VIRL. Co se týče porovnání paměťových a výpočetních nároků GNS3 podle nasazení OS Windows a Linux, zde dosahuje lepších výsledků operační systém Linux na menších topologiích. Na rozsáhlém scénáři 1 bylo vidět, že v operačním systému Linux jsou výpočetní nároky větší, než na operačním systému Windows.

V testech zaměřených na přenosovou rychlost v rámci jednotlivých scénářů nejlépe dopadlo simulační prostředí GNS3 nasazené na operačním systému Linux. V kategorii testů zaměřených na ztrátovost UDP datagramů během přenosu nejlépe obstálo simulační prostředí GNS3 na operačním systému Windows. Při testování TCP spojení se nedá říct, které prostředí je lepší, jestli VIRL, nebo GNS3. Naopak můžu tvrdit, že ze získaných hodnot během testování TCP spojení se naměřené hodnoty při použití GNS3 na operačním systému Linux a Windows výrazně neliší.

Profesionální řešení VIRL dominovalo v testech zaměřených na rychlost odezvy. Simulační prostředí GNS3 při testu odezvy dosahovalo lepších výsledků na operačním systému Linux, než Windows. Z hlediska konfigurovatelnosti je na tom lépe profesionální řešení VIRL. Tato vlastnost je dána faktem, že VIRL podporuje většinu technologií které známe ze zařízení nabízené firmou Cisco, právě proto, že VIRL je nabízen jako komerční produkt právě touto firmou.

Použitím simulačního prostředí GNS3 uživatel získá lepší přenosovou rychlost, menší ztrátu při přenosu, menší výpočetní a paměťové nároky pro svou práci, ale za cenu omezených možností konfigurace. Naopak použitím profesionálního řešení VIRL uživatel získá nástroj, který umožňuje provádět konfigurace většiny technologií potřebných pro získání certifikátů od firmy Cisco.

Hlavním přínosem pro mne samotného z této diplomové práce jsou získané poznatky během testování obou simulačnách prostředít a také možnost vyzkoušet si práci v profesionálním řešení VIRL od firmy Cisco.

V budoucnu by bylo možné rozšířit tuto práci o další scénáře. Vhodná oblast pro tvorbu nových scénářů by mohla být bezpečnost počítačových sítí. Zajímavé by bylo nad těmito scénáři z oblasti bezpečnosti počítačových sítí provádět testy pomocí linuxové distribuce Kali. Dále by bylo možné nasadit a otestovat navržené scénáře na čisté platformě OpenStack. Vzhledem k získaným hodnotám při testování paměťové náročnosti jednotlivých scénářů v simulačním prostředí GNS3 by bylo dobré zaměřit se na rozptýlení pochyb ohledně programu dynamips na operačním systému Linux.

Literatura

[1] CODY, V. K. *OpenStack in Action*. United States of America: Manning Publications Co., 2016. ISBN 9781617292163.

[2] OpenStack diagram. In: *OpenStack* [online]. [cit. 2017-04-23]. Dostupné z: https://www.openstack.org/themes/openstack/images/software/openstack-software-diagram.png

[3] *VIRL tutorial* [online]. 2017 [cit. 2017-04-24]. Dostupné z: http://virl-dev-innovate.cisco.com/tutorial.php

[4] HW requirements. *Cisco VIRL* [online]. 2017 [cit. 2017-04-24]. Dostupné z: http://virl.cisco.com/getvirl/

[5] Installing VIRL on VMware Workstation. *Cisco VIRL* [online]. 2017 [cit. 2017-04-24]. Dostupné z: http://virl-dev-innovate.cisco.com/workstation.php

[6] Node Types. *The Packet Thrower's Blog* [online]. [cit. 2017-04-24]. Dostupné z: https://the-packet-thrower.com/ccna-workbook-rs/network-fundamentals/using-virl/

[7] NEUMANN, Jason C. *The Book of GNS3: Build Virtual Network Labs Using Cisco, Juniper, and More*. San Francisco, 2015. ISBN 1-59327-554-4.

[8] About GNS3 Technologies Inc. In: *GNS3 Technologies Sets Crowd-funding Record* [online]. 2014 [cit. 2017-04-24]. Dostupné z: http://www.thecollective1212.space/blog/2014/9/2/gns3-technologies-sets-crowdfunding-record

[9] WELSH, Chris. *GNS3 Network Simulation Guide*. BIRMINGHAM: Packt Publishing, 2013. ISBN 978-1-78216-080-9.

[10] CPU Benchmarks. *PassMark* [online]. [cit. 2017-04-25]. Dostupné z: *www.cpubenchmark.net/cpu_list.php*

[11] WARNER, Timothy. *PowerShell Get-Process - Managing processes* [online]. [cit. 2017-04-24]. Dostupné z: https://4sysops.com/archives/powershell-get-process-managing-processes/

A Obsah přiloženého CD

Součást diplomové práce je přiložené CD, které obsahuje následující soubory:

1. Diplomovou práci ve formátu pdf.
2. Adresář Projekty, ve kterém se nachází spustitelné projekty jednotlivých scénářů pro prostředí VIRL a GNS3.
3. Adresář Konfigurace, který obsahuje konfigurační soubory jednotlivých scénářů.
4. Adresář Zachycení provozu, který obsahuje zachycené záznam komunikace ve formátu .pcap pro prostředí VIRL a GNS3.
5. Adresář Paměťové a výpočetní nároky, který obsahuje soubory ve formátu .xlsx na kterých jsou kompletní naměřené hodnoty pro jednotlivé testy paměťových a výpočetních nároků.